A PROJECT REPORT ON

# "ARCHITECTURAL FRAMEWORK FOR VIRTUAL COMPUTING LAB"

BY

| | |
|---|---|
| AMIT PANDIT | B80384239 |
| NISHANT GUPTA | B80384273 |
| VISHAL GAURAV | B80384230 |
| ANIKET TRIVEDI | B80384205 |

Under the Guidance of

Mrs.SUKHADA BHINGARKAR

In fulfillment of

B.E. (COMPUTER)

DEGREE OF UNIVERSITY OF PUNE,

MAY/JUNE 2014

DEPARTMENT OF COMPUTER ENGG.

M.I.T. COLLEGE OF ENGINEERING

PUNE – 411 038.

**MIT COLLEGE OF ENGINEERING**
DEPARTMENT OF COMPUTER ENGINEERING
PUNE – 411038

# *CERTIFICATE*

This is to certify that the project report entitled

**"ARCHITECTURAL FRAMEWORK FOR VIRTUAL COMPUTING LAB"**

**Submitted by**

| ***Group Members*** | ***Seat No's*** |
|---|---|
| **AMIT PANDIT** | **B80384239** |
| **NISHANT GUPTA** | **B80384273** |
| **ANIKET TRIVEDI** | **B80384205** |
| **VISHAL GAURAV** | **B80384230** |

is a record of bonafide work carried out by them, under my guidance, in fulfillment of the requirement for the award of Degree of Bachelor of Engineering(Computer) at M.I.T. College of Engineering, Pune under University of Pune.

Date: /04/2014

Place: MITCOE, Pune

**Mrs.Sukhada Bhingarkar**
**Guide,**
**Dept.of Computer Engineering,**
M.I.T. College of Engineering
**Pune – 411038**

**Prof.Dr. Prasanna Joeg**
**Head,**
**Dept.of Computer Engineering**
M.I.T. College of Engineering
**Pune – 411038**

The Project entitled

# “ARCHITECTURAL FRAMEWORK FOR VIRTUAL COMPUTING LAB”

***BY***

| | |
|---|---|
| **AMIT PANDIT** | **B80384239** |
| **NISHANT GUPTA** | **B80384273** |
| **VISHAL GAURAV** | **B80384230** |
| **ANIKET TRIVEDI** | **B80384205** |

is approved for the degree of

**BACHELOR OF ENGINEERING – COMPUTER**

**University of Pune, Pune.**

**Examiners:** 1. ________________________________________

2. ________________________________________

**Date:**

**Place:**

# ACKNOWLEDGEMENT

Apart from efforts taken by us, the completion , the success of this project depends largely on the encouragement , motivation and guidelines of many others .We take this opportunity to express our gratitude to all the people who have been a big part in the completion of this project.

We are also very thankful to our Principal,**Prof.Dr.Vijay Wadai** and **Prof.Dr.Prasanna Joeg**, H.O.D. Dept. of Computer Engineering for providing us with such a great opportunity.

We are very grateful to our project guide **Mrs.Sukhada Bhingarkar,** for her constant guidance and whole hearted support and much valued constructive criticism that has been a driving force to complete this project successfully. We felt motivated and encouraged every time we attended the meeting. Without her encouragement, guidance, devotion of time and help, this project would not have materialized.

We are also thankful to **Mrs.Bharti Ainapure** for motivating us and providing us guidelines for the completion of the project.

Last but not the least; we would like to thank our parents for constantly encouraging us .We would also like to take this opportunity to thank our friends, who were always a source of encouragement.

| | |
|---|---|
| Amit Pandit | B80384239 |
| Nishant Gupta | B80384273 |
| Aniket Trivedi | B80384205 |
| Vishal Gaurav | B80384230 |

# CONTENTS

**Page No.**

**Abstract**

**List of Tables**

**List of Figures**

# Abstract

In private cloud computing, we need to make sure that large number of computing resources are available on demand, however in current architectures like eucalyptus, there is a single server or cluster controller (CC) which is a sole provider of virtual machines to the users. Using eucalyptus, we can build a private cloud, through which an educational organization can build a virtual computing lab that can satisfy the requirements of an organization. But, due to extensive use of computational resources, cluster controller component of eucalyptus becomes a bottleneck, hampering performance. Also, we have studied different drawbacks associated with different architectures. The proposed architecture overcomes the most drawbacks. This architecture for virtual cloud computing is based on a master –slave concept where one master and multiple salves serve the resources to the clients. This approach improves the performance of the server and would allow cloud servers to extend their computation power by dynamic resource discovery over the network. It allows new clients to request virtual machines, and the server makes the choice of running the requested virtual machine either on previously available slaves, or on the clients who are recently registered into a set of slaves. Along with this dynamically growing architecture, we have also implemented an algorithm used to schedule the requests coming from different clients based on the priority which are calculated through various parameters. Using this algorithm, our architecture resolves various problems, race in conditions and priority requests. Thus, this architecture reduces the probability of occurrence of network bottlenecks and ensures that sufficient resources are always available to the end users, thus implementing the concept "Cloud never dies".

**Keywords:** Cloud Computing, Master, Private Cloud, Slave, Virtual Machine, Para-Scheduling algorithm, Priority.

# Chapter I

# INTRODUCTION

## 1.1 Cloud Computing

Cloud Computing is a technology that uses the internet and central remote servers to maintain data and applications. Cloud computing allows consumers and businesses to use applications without installation and access their personal files at any computer with internet access. This technology allows for much more efficient computing by centralizing data storage, processing and bandwidth.

A simple example of cloud computing is Yahoo email, Gmail, or Hotmail etc. All you need is just an internet connection and you can start sending emails. The server and email management software is all on the cloud (internet) and is totally managed by the cloud service provider Yahoo, Google etc. The consumer gets to use the software alone and enjoy the benefits [5].

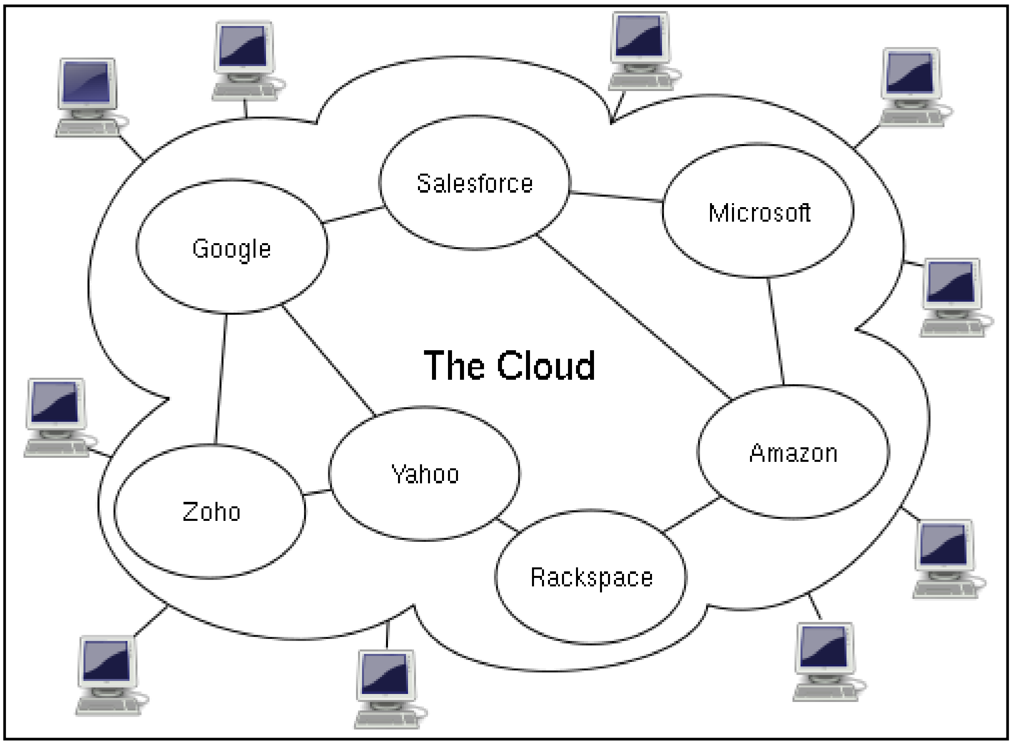


**Figure 1: Simple Cloud Architecture**

## 1.2 Types of Cloud

The cloud isn't a technology. It's more of an approach to building IT services - an approach that harnesses the power of servers, as well as virtualization technologies that combine servers into large computing pools and divide single servers into multiple virtual machines. And there are several different deployment models for implementing cloud technology [6].

The four primary types of cloud models are:

- Public
- Private
- Hybrid
- Community

Each has its advantages and disadvantages with significant implications for any organization researching or actively considering a cloud deployment.

**Public Cloud**

A public cloud is a cloud computing model in which services, such as applications and storage, are available for general use over the Internet. Public cloud services may be offered on a pay-per-usage mode or other purchasing models. An example of a public cloud is IBM's Blue Cloud.

**Private Cloud**

A private cloud is a virtualized data center that operates within a firewall. Private clouds are highly virtualized, joined together by mass quantities of IT infrastructure into resource pools, and privately owned and managed.

**Hybrid Cloud**

A hybrid cloud is a mix of public and private clouds.

**Community Cloud**

A community cloud is an infrastructure shared by several organizations which supports a specific community.

## 1.3 Cloud Services

The following 3 descriptions are being used in the industry today to describe the main types of cloud computing service offerings

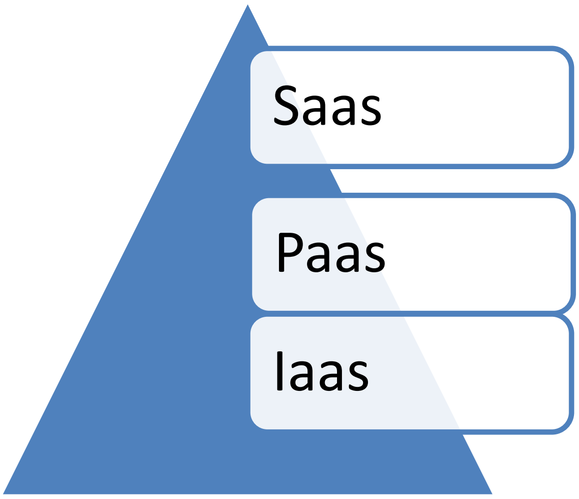


**Figure 2: Cloud Services**

**Software as a Service (SaaS)**

At the highest level is **Software as a Service**. SaaS delivers an application to a consumer or business user through a web browser client. The business logic and data for the application run on a server living somewhere on the network, not through an application running on the user's computer. The software is normally sold to the end user via a subscription, as opposed to a one time, upfront license fee [6].

There are 1000s of examples of SaaS applications. One of the first, and still one of the best known, is Salesforce.com, which is an enterprise CRM tool. Other popular examples of SaaS applications are:

- Box.net
- Google Docs
- Microsoft Office 365
- Jira
- Basecamp

### Infrastructure as a Service (IaaS)

At the lowest level is **Infrastructure as a Service**. IaaS is as close to the “metal” as you can get. An IaaS service provider typically provides networked computers running in a hosted environment. At it simplest, the IaaS provides the hardware and, usually virtualized, OS.

Every web hosting company out there could almost be considered an IaaS provider. The key differentiator between a web hosting provider and an IaaS provider is how they charge for their services. A web hosting company charges by the system by the month. An IaaS provider charges only for the compute power that is utilized (usually by CPU hours used by month).

By this definition, some of the more well known IaaS providers are:

- Amazon EC2 (elastic cloud compute)
- Rackspace
- Google Compute Engine

### Platform as a Service (PaaS)

Somewhere in the middle is **Platform as a Service** (PaaS). The biggest difference between IaaS and PaaS is that PaaS adds support for the development environment (development language and application server technology).

By writing your application in this environment you can very easily take advantage of dynamic scalability, automated database backups, and other platform services without the need to specifically code for it. For this reason, PaaS offerings generally support a specific set of programming languages or development environments.

PaaS services are usually billed as an incremental cost on top of the IaaS monthly charges. For example, there may be a small monthly fee for the use of a load balancer or a database backup service [6].

## 1 .4 what is Virtual Computing?

Virtual computing allows computer users remote access to software applications and processes when they need it. Users gain access via the Internet through a wireless or network server. For a fee, users can boost their computers' capabilities, size, performance, processes and/or software applications whenever they need it [7].

This real-time technology offers:

- Operating and utility systems
- Storage
- Memory
- Software
- Allocation and reassignment of input/output and other processes
- Data backup
- Automated problem solving and troubleshooting
- Tools for monitoring and managing systems

Users can access software applications for a single computer or an entire network because of the ability to select only what you need when you need it. They also can save or back up data and text documents to a virtual server (thus freeing space on individual computers) and reallocate or assign different processes to the virtual environment. This enables computers to operate at optimal speeds.

Virtual computing initially began as a method of borrowing space or storage for computer systems, but it's since grown significantly, offering data and software applications, as well as operating and utility systems. The corporate environment most commonly uses it, where IT system managers run multiple applications on several servers.

## 1.5 Virtualization

Virtualization, in computing, refers the act of creating a virtual (rather than actual) version of something, including but not limited to a virtual computer hardware platform, operating system (OS), storage device, or computer network resources.

The term "virtualization" traces its roots to 1960s mainframes, during which it was a method of logically dividing the mainframes' resources for different applications [7].

**Server Virtualization:** The architecture of today's x86 servers allows them to run only one operating system at a time. Server virtualization unlocks the traditional one-to-one architecture of x86 servers by abstracting the operating system and applications from the physical hardware, enabling a more cost-efficient, agile and simplified server environment. Using server virtualization, multiple operating systems can run on a single physical server as virtual machines, each with access to the underlying server's computing resources.

Server virtualization unleashes the potential of today's powerful x86 servers. Most servers operate less than 15 percent of capacity; not only is this highly inefficient, it also introduces server sprawl and complexity.

**Network Virtualization:** Network virtualization is the complete reproduction of a physical network in software. Virtual networks offer the same features and guarantees of a physical network, yet they deliver the operational benefits and hardware independence of virtualization rapid provisioning, non disruptive deployment, automated maintenance and support for both legacy and new applications.

**Desktop Virtualization:** Deploying desktops as a managed service gives you the opportunity to respond quicker to changing needs and opportunities. You can reduce costs and increase service by quickly and easily delivering virtualized desktops and applications to branch offices, outsourced and offshore employees and mobile workers on iPad and Android tablets.

# Chapter II

# LITERATURE SURVEY

## 2.1 Study of Different types of Architectures

### 2.1.1 Eucalyptus:

Eucalyptus is the only platform to offer a hierarchical structure. There is an interface, known as the Client API, which enables communication and connection between each of the resources on the cloud computing platforms. Three placement policies are pre-defined: round robin, greedy and green -it. All policies are static, i.e. the placement decisions are made only for new requests. There is no migration of instances during runtime. Also, due to extensive usage of computational resources, cluster controller (CC) component of Eucalyptus becomes a bottleneck, hampering performance of cloud computing environment.

Eucalyptus is designed to be an open-source answer to the commercial EC2 cloud. Eucalyptus, for its frontend for users, includes a program called euca2ools, which is very similar to Amazon's EC2 front-end programs. However, some versions of EC2 itself, as well as Right scale are also compatible front-ends. The overall design specification of the cloud has been published [4].

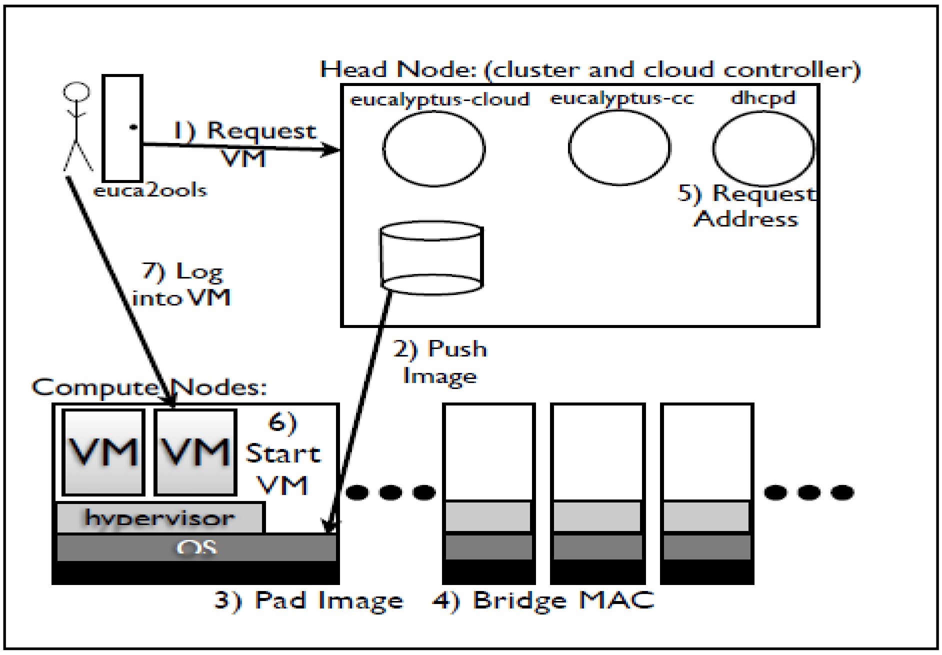


**Figure 3 .Eucalyptus [4]**

**The steps for constructing a virtual machine in a configuration of Eucalyptus:**

1) A user uses the euca2ools front-end to request a VM.

2) The VM template disk image is pushed to a compute node.

3) This disk image is padded to the correct size and packaged for use by the hypervisor on the compute node.

4) The compute node sets up network bridging to provide a virtual NIC with a virtual MAC.

5) On the head node the dhcp is set up with the MAC/IP pair. (Note: Technically this is STATIC mode. But other modes are similar.

6) VM is spawned on the VMM.

7) The user can now SSH directly into the VM along with an overview of the most recent version, and much of the feature documentation is available online.

In figure 3, we give the steps taken in spawning a VM in a typical installation. With regard to the overall philosophy underlying Eucalyptus, the system is very much designed for the corporate enterprise computing setting, a fact confirmed by the language used on its website and in its documentation.

Table 1.Eucalyptus

| **Advantages** | **Disadvantages** |
|---|---|
| Only platform to offer hierarchical structure | Placement decisions are made only for few requests i.e. are static |
| Open-Source | Extensive use of computational resources |
| Full API compatibility with Amazon EC2 | No migration of instances during runtime. Due to which cluster controller component becomes bottleneck |
| Placement policies are pre-defined | Lack of single sign-on support for its web interface and the heavyweight java implementation |

### 2.1.2 Open Nebula:

Open Nebula provides the **most simple but feature-rich and flexible solution** for the comprehensive management of virtualized data centers to enable on-premise IaaS clouds. Open Nebula interoperability makes cloud an evolution by leveraging existing IT assets, protecting your investments, and avoiding vendor lock-in.

Open Nebula can be primarily used as a platform to manage your virtualized infrastructure in the data center or cluster, which is usually referred as **Private Cloud**. Open Nebula supports **Hybrid Cloud** to combine local infrastructure with public cloud-based infrastructure, enabling highly scalable hosting environments. Open Nebula also supports **Public Clouds** by providing Cloud interfaces to expose its functionality for virtual machine, storage and network management.

Open Nebula tends to a greater level of centralization and customizability (especially for end-users).

The steps for spawning a VM are shown in Figure 4. The configuration in this figure is the default configuration which is documented on their website. However, there is a huge amount of customizability that is permitted in Open Nebula. Specifically, the idea of Open-Nebula is a pure private cloud, in which users actually log into the head node to access cloud functions. This interface is a wrapper around an XML-RPC interface, which can also be used directly. We note that a frontend interface, such as EC2, can be appended to this default configuration [4].

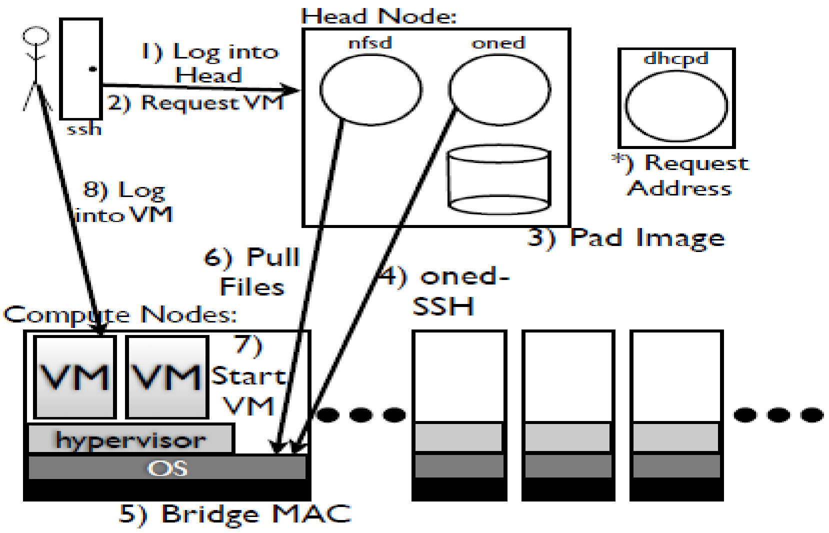


**Figure 4.Open-Nebula [4]**

**The steps for constructing a VM in a configuration of Open Nebula:**

1) A user uses ssh to login to the head node.

2) The user uses the one vm command to request a VM.

3) The VM template disk image is copied and a copy is padded to the correct size and configuration within the NFS directory on the head node.

4) The oned process on the head node uses ssh to log into a compute node.

5) The compute node sets up network bridging to provide a virtual NIC with a virtual MAC.

6) Files needed by the VMM on the compute node will be pulled to the compute node via the NFS.

7) VM is spawned on the VMM.

8) The user can now SSH directly into the VM.

9) Typically, the dhcp server is handled separately from the Open Nebula configuration.

Table 2.Open-Nebula

| **Advantages** | **Disadvantages** |
|---|---|
| Able to convert a physical computer cluster into a flexible, virtual infrastructure. | Lacks the capability to integrate into an existing authentication infrastructure such as Kerberos |
| In the event of malfunction, the background can be restarted and VM’s can be restored. | Scheduling problem |
| Architecture allows introduction of new policies | Have very restricted monitoring feature |

### 2.1.3 OpenStack:

Open Stack offers open source software to build public and private clouds. OpenStack Compute is a cloud fabric controller, used to start up virtual instances for either a user or a group. It's also used to configure networking for each instance or project that contains multiple instances for a particular project. But OpenStack Compute also suffers by the problem of network bottleneck in a situation of high demand.

**OpenStack** is a global collaboration of developers and cloud computing technologists producing the ubiquitous open source cloud computing platform for public and private clouds. The project aims to deliver solutions for all types of clouds by being simple to implement, massively scalable, and feature rich. The technology consists of a series of interrelated projects delivering various components for a cloud infrastructure solution [9].

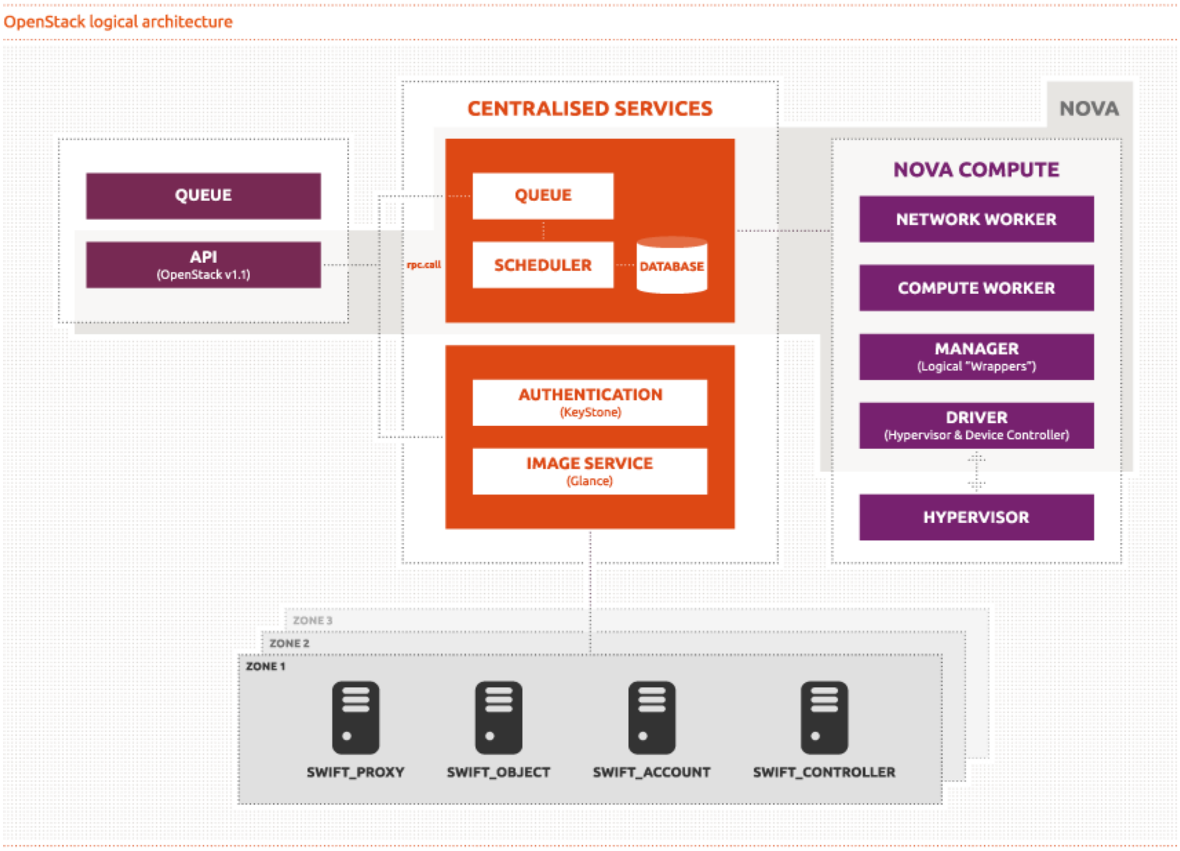


**Figure 5: Architecture for OpenStack [9]**

Table 3.Open-Stack

| Advantages | Disadvantages |
|---|---|
| Open-source software to build public and private clouds. | Suffers from a problem of bottleneck in a situation of high demand. |
| Used to configure networking for each instances | Servers are not always reliable and issues cloud dissatisfy customers |
| Allows to start VM's for a user or group of users | Is not compatible with multi-languages or multi-currency |

### 2.1.4 Nimbus:

Nimbus is an Open Source toolkit which is also particularly suitable for creating a community cloud. It consists of many different function components to support the various modules on the platform. Commands and results are entered in command lines, which make Nimbus appear to be very complex because of the specific applications. Also, a negative factor is the lack of support for VMware.

The Nimbus project explicitly advertises itself as a "science" cloud solution. Nimbus is also affiliated with the Globus Project and uses Globus credentials for the authentication of users. Until recently, Nimbus relied on GridFTP (a Globus project) to be used as a disk image repository. In their newest version, they are shifting to Cumulus, a system, like Eucalyptus' Walrus, compatible with S3. Like Open- Nebula, Nimbus is incredibly customizable. Figure 4 shows the steps in spawning a VM in a "typical" configuration [4].

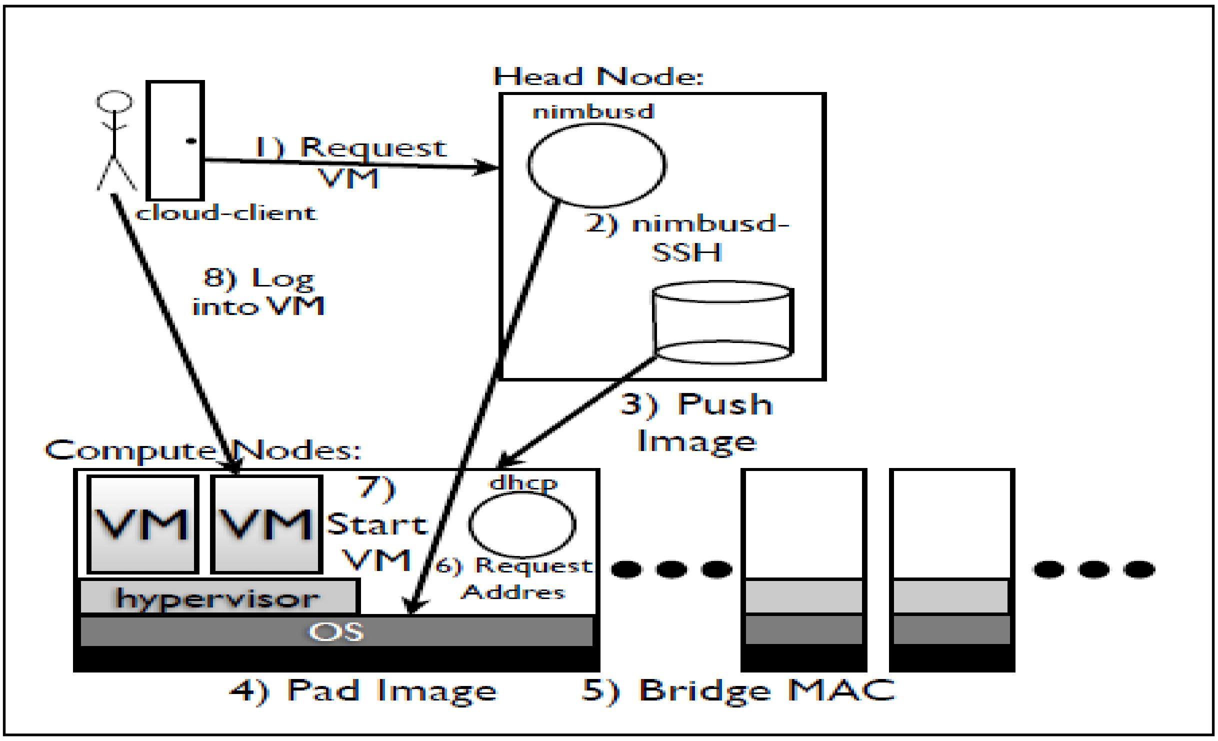


**Figure 6 .Nimbus [4]**

**The steps for constructing a virtual machine in a configuration of Nimbus:**

1) A user uses cloud-client to request a VM.

2) Nimbus will ssh into a compute node.

3) The VM template disk image is pushed to the compute node. (In the newest versions of Nimbus, this will be done using a distributed storage similar to S3 and Walrus.)

4) On the compute node, the disk image is padded to the correct size and configured.

5) The compute node sets up network bridging to provide a virtual NIC with a virtual MAC.

6) A dhcp server on the compute node is configured with a MAC/IP pair.

7) VM is spawned on the VMM. 8) The user can now SSH directly into the VM.

Table 4.Nimbus

| Advantages | Disadvantages |
|---|---|
| Suitable for creating a community cloud | Commands and results are entered in command lines, which makes nimbus appear to be very complex |
| Consists of different function components to support various modules on platform | Lack of support of VMware. |

Table 5.Cloud Architectures Compared

| | Eucalyptus | OpenNebula | Nimbus |
|---|---|---|---|
| Disk Image Options | Options set by admin | In private cloud, most libvirt options left open. | Depends on configuration |
| Disk Image Storage | Walrus, which imitates Amazons S3 | A shared file system, by default NFS, or SCP | Cumulus (recent update from GridFTP) |
| Hypervisors | Xen,KVM(VM Ware in non-open source) | Xen,KVM,Vmware | Xen,KVM |
| Unique Features | User management web interface | VM migration supported | Nimbus context broker |

## 5] Ovirt

oVirt is a free Red Hat / Fedora specific virtualization management system that allows to manage virtual machines through a web interface using libvirt. However, it is unable to scale to external clouds. It has monolithic and closed architecture that is hard to extend or interface with other software, not allowing seamless integration with existing storage and network management solutions deployed in data centers.

Table 6.Ovirt

| Advantages | Disadvantages |
|---|---|
| Allows to manage VM's through web interface | Unable to scale to external clouds. |
| oVirt can manage multiple hosts | Has monolithic and closed architecture which is hard to extend or interface with other software. |

## 2.2 Study of Eucalyptus in detail

Eucalyptus is a free and open-source computer software for building Amazon Web Services (AWS)-compatible private and hybrid cloud computing environments marketed by the company Eucalyptus Systems. Eucalyptus enables pooling compute, storage, and network resources that can be dynamically scaled up or down as application workloads change

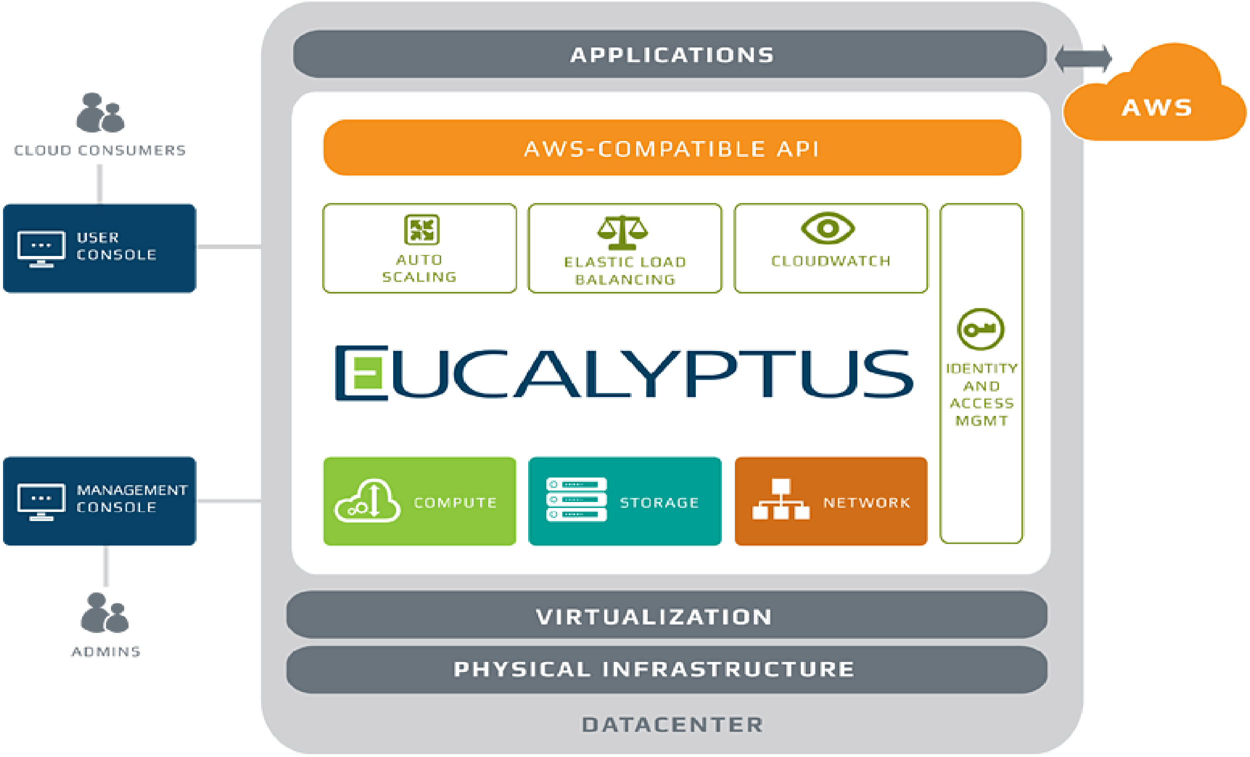


**Figure 7.Architecture Overview [10]**

**Components:**

Eucalyptus has six components:

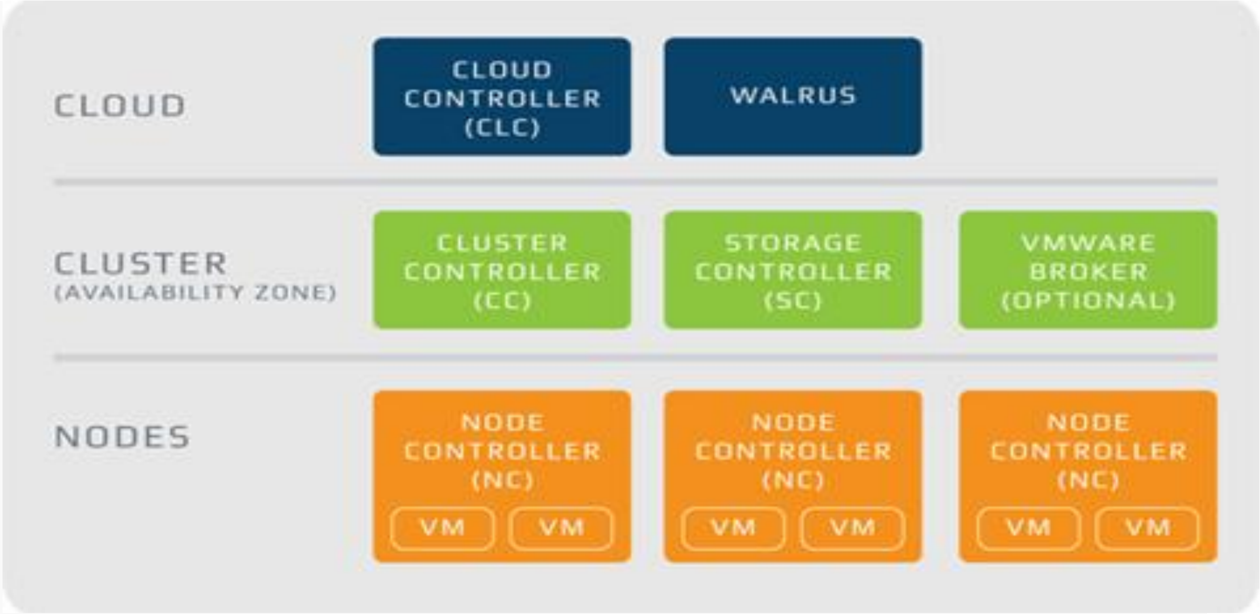


**Figure 8. Components of Eucalyptus [10]**

- The ***Cloud Controller (CLC)*** is a Java program that offers EC2-compatible interfaces, as well as a web interface to the outside world. In addition to handling incoming requests, the CLC acts as the administrative interface for cloud management and performs high-level resource scheduling and system accounting. The CLC accepts user API requests from command-line interfaces like euca2ools or GUI-based tools like the Eucalyptus User Console and manages the underlying compute, storage, and network resources. Only one CLC can exist per cloud and it handles authentication, accounting, reporting, and quote management.

- ***Walrus,*** also written in Java, is the Eucalyptus equivalent to AWS Simple Storage Service (S3). Walrus offers persistent storage to all of the virtual machines in the Eucalyptus cloud and can be used as a simple HTTP put/get storage as a service solution. There are no data type restrictions for Walrus, and it can contain images (i.e., the building blocks used to launch virtual machines), volume snapshots (i.e., point-in-time copies), and application data. Only one Walrus can exist per cloud.

- The ***Cluster Controller (CC)*** is written in C and acts as the front end for a cluster within a Eucalyptus cloud and communicates with the Storage Controller and Node Controller. It manages instance (i.e., virtual machines) execution and Service Level Agreements (SLAs) per cluster.

- The ***Storage Controller (SC)*** is written in Java and is the Eucalyptus equivalent to AWS EBS. It communicates with the Cluster Controller and Node Controller and manages Eucalyptus block volumes and snapshots to the instances within its specific cluster. If an instance requires writing persistent data to memory outside of the cluster, it would need to write to Walrus, which is available to any instance in any cluster.

- The ***VMware Broker*** is an optional component that provides an AWS-compatible interface for VMware environments and physically runs on the Cluster Controller. The VMware Broker overlays existing ESX/ESXi hosts and transforms Eucalyptus Machine Images (EMIs) to VMware virtual disks. The VMware Broker mediates interactions between the Cluster Controller and VMware and can connect directly to either ESX/ESXi hosts or to vCenter Server.

- The ***Node Controller (NC)*** is written in C and hosts the virtual machine instances and manages the virtual network endpoints. It downloads and caches images from Walrus as well as creates and caches instances. While there is no theoretical limit to the number of Node Controllers per cluster, performance limits do exist [10].

**Functionality provided:**

The Eucalyptus User Console provides an interface for users to self-service provision and configures compute, network, and storage resources. Development and test teams can manage virtual instances using built-in key management and encryption capabilities. Access to virtual instances is available using familiar SSH and RDP mechanisms. Virtual instances with application configuration can be stopped and restarted using encrypted boot from EBS capability.

IaaS service components Cloud Controller, Cluster Controller, Walrus, Storage Controller, and VMware Broker are configurable as redundant systems that are resilient to multiple types of failures. Management state of the cloud machine is preserved and reverted to normal operating conditions in the event of a hardware or software failure.

Eucalyptus can run multiple versions of Windows and Linux virtual machine images. Users can build a library of Eucalyptus Machine Images (EMIs) with application metadata that are decoupled from infrastructure details to allow them to run on Eucalyptus clouds. Amazon Machine Images are also compatible with Eucalyptus clouds. VMware Images and vApps can be converted to run on Eucalyptus clouds and AWS public clouds.

Eucalyptus user identity management can be integrated with existing Microsoft Active Directory or LDAP systems to have fine-grained role based access control over cloud resources.

Eucalyptus supports storage area network devices to take advantage of storage arrays to improve performance and reliability. Eucalyptus Machine Images can be backed by EBS-like persistent storage volumes, improving the performance of image launch time and enabling fully persistent virtual machine instances. Eucalyptus also supports direct-attached storage [10].

## 2.3 Problem Statement

Virtual Computing Lab is intended for designing and configuring a private cloud computing system that serve both the educational and research missions in as efficient manner.

The VCL is developed using eucalyptus which has several features. But, it suffers from certain drawbacks e.g. to serve 48 students in a laboratory simultaneously, we had setup a server with 2 Xeon 6 core each processor and 48 GB RAM. But in a college, out of 700 machines, at least 250 machines are required to run virtual machine instances for performing Local Area Network. This limits the number of resources and thus the perception of infinite resources in a private cloud fails.

Thus, this leads to design an architecture which ensures that sufficient resources are always available and the cloud never dies.

# Chapter III

# PROPOSED ARCHITECTURE

## 3.1 Need of Proposed Architecture

In private cloud computing, we need to make sure that large number of computing resources are available on demand, however in current architectures like eucalyptus, there is a single server or cluster controller (CC) which is a sole provider of virtual machines to the users. Using eucalyptus, we can build a private cloud, through which an educational organization can build a virtual computing lab that can satisfy the requirements of an organization. But, due to extensive use of computational resources, cluster controller component of eucalyptus becomes a bottleneck, hampering performance.

Also, we have studied different drawbacks associated with different architectures. The proposed architecture overcomes the most drawbacks.

This architecture for virtual cloud computing is based on a master –slave concept where one master and multiple salves serve the resources to the clients. This approach improves the performance of the server and would allow cloud servers to extend their computation power by dynamic resource discovery over the network. It allows new clients to request virtual machines, and the server makes the choice of running the requested virtual machine either on previously available slaves, or on the clients who are recently registered into a set of slaves.

Thus, this architecture reduces the probability of occurrence of network bottlenecks and ensures that sufficient resources are always available to the end users, thus implementing the concept “Cloud never dies” [2].

## 3.2 Solutions to be provided

To solve the problem of scalability and limitation in resources, this paper proposes an architecture which would not only utilize the resources of the cloud server, but would also be capable of running lesser priority virtual machines on client machines running our specialized client OS. The adaptive cloud is based on master-slave approach which consists of one master and one slave at the beginning. First of all, the slave has to register itself to the master to be part of cloud. After the registration with the master, the master queries the slave about the available resources. The slave replies to the master and master simultaneously updates its database about the number of cores it has. All the images are bundled and uploaded to the master so that they can be launched on a slave. This architecture grows dynamically when new clients are added to a set of slaves.

This architecture allows new clients to request virtual machines, and the server makes the choice of running the requested virtual machine either on previously available slaves, or on the clients who are recently registered into a set of slaves. It allows setting up node controller functionalities on machines over network that increases the capability to run more virtual instances.

The proposed solution also provides basic cloud computing features like launching virtual instances, keeping track of status of virtual machines, which are running on node controller, addressing of these virtual machines, NATing and mapping of virtual machines

## 3.3 Proposed Architecture and its features

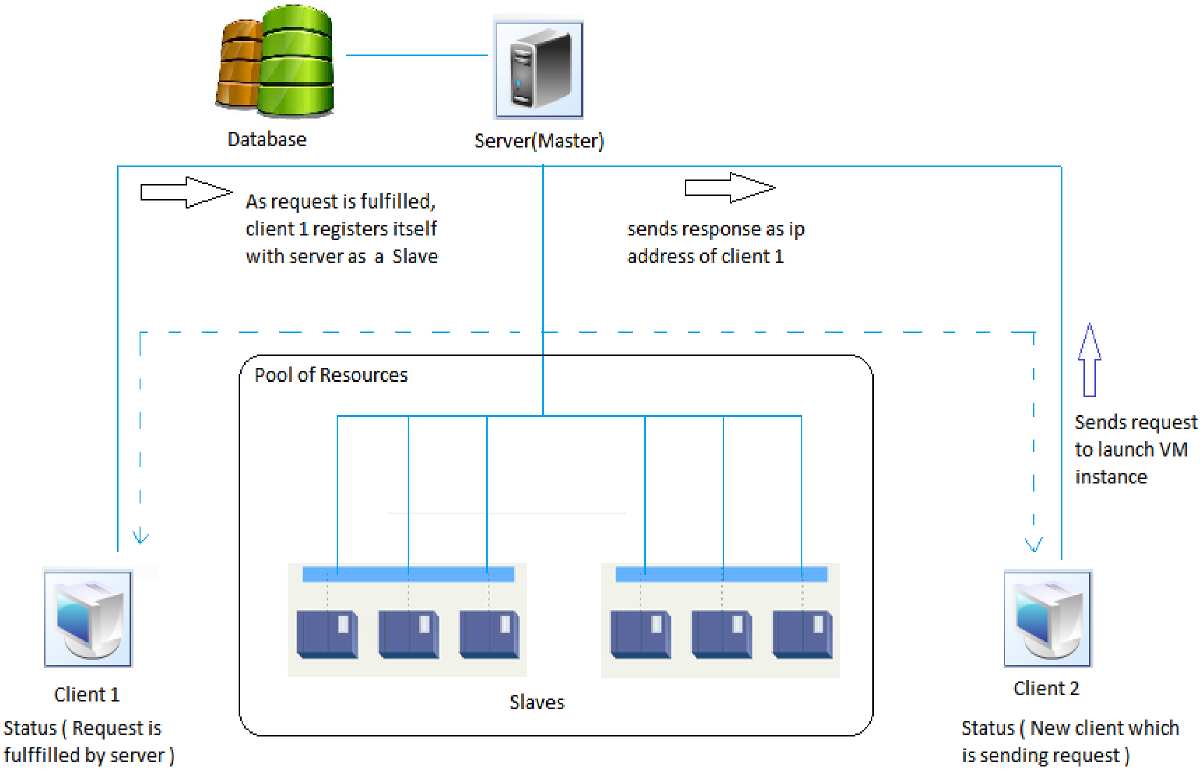

Database
Server(Master)
As request is fulfilled, client 1 registers itself with server as a Slave
sends response as ip address of client 1
Pool of Resources
Sends request to launch VM instance
Client 1
Status ( Request is fulfilled by server )
Slaves
Client 2
Status ( New client which is sending request )

**Figure 9.Proposed Architecture**

The architecture consists of following components as shown in figure 2:

**Client1**: This machine sends request to the master machine. Corresponding response is generated showing the status of VM and IP address of VM. If the status is “running”, client is connected to VM instance using specified IP address.

**Client2:** It is other client machine that wants to launch VM instance.

**Server (Master):** It is responsible to fulfill the requests of clients for launching the VM instances. Whenever any client machine sends request to the server machine for VM instance, corresponding response is sent back to the client specifying status of VM instance and IP address of requested VM instance so as to connect with the VM.

**Slaves**: It is a pool of resources. When client request is fulfilled, client registers itself with server machine. Client machine announces its resources with server machine and server adds client machine in the resource pool and monitors its status.

## 3.4 Proposed Para-Scheduling Algorithm

To solve the problem of scalability and limitation in resources, this paper proposes a Para-Scheduling algorithm which would not only utilize the resources of the cloud master, but would also provide an efficient way to schedule the request having greater priority. Every request has various parameters like type of resource demanded, amount of resource required, time for which the resource is required. Considering all the parameters, it requires an algorithm, which should provide the resources to the client in an efficient manner. Also, the need to maintain at least resource available in the pool of slaves should be satisfied by the algorithm.

If such efficiency is not achieved, then there could be situation where even the resources is available but cannot be allocated to the client. In order to overcome this efficient resource utilization problem, the algorithm is to be used in the architecture.

Here .

$R_i$ : is request id.

$A_i$ : is i'th client .

S : is set of slaves.

1. Client sends a request to master with the required information.

2. Master have list of requests $R_i$ of client $A_i$ .

3.1.i) If there's only one request then master checks the availability of resources.

ii) If the resource is available then a connection is established between current client and the slave.

3.2. i) If the resource is not available then the master checks the priority of a client using the resource and the client demanding the resource.

ii) One having greater priority get access to resource.

iii) If both have same priority then the demanding client has to wait until the resource is free.

4. i) If there are multiple requests, then

ii) First of all priority of the entire request is calculated.

iii) If two or more requests have same priority then they are scheduled on first come first serve basis.

iv) Next step is checking the availability of resources which is same as for one request.

5. After permission checking, availability of requested resource is checked

i) If the resource is available then it is assigned to the client

ii) Else given resource is installed and then assigned to the client.

6. When clients request is fulfilled, the resource is released and database is updated.

7. Now if the client wants to register itself as a slave it is registered and S is updated as

$$S = \sum Sk \cup ai$$

8. While there are unprocessed request go to step 3.

9. End

The representation of algorithm by flowchart:

## 3.5 Platform/Technology

According to proposed architecture we need following setup to deploy our architecture.

1. Software interface for user on the client machine will be his/her computer having ubuntu as its operating system.
2. The hardware should provide Virtualization Technology and other necessary compatibility.
3. The ubuntu machine should be running TightVNCViewer
4. Web Interface is provided for users to launch a VM.
5. Tight VNC Viewer manages all UI related with running the VM.
6. The Master is connected to a DHCP Server.

Table 7 Presents machine configuration for the private cloud.

| Sr.No | Machine Configuration | Role |
|---|---|---|
| 1. | 64-bit Ubuntu 9.10 server with 4 GB RAM | Master |
| 2. | 32-bit/64-bit Ubuntu 9.10 desktop with 4 GB RAM having VT enabled. | Client 1 |
| 3. | 32-bit/64-bit Ubuntu 9.10 desktop with 2 GB RAM<br>Having VT enabled. | Client 2 |

**The assumptions for this architecture are as follows:**

1. This architecture solely focuses on to check the possibility whether cloud solution can extend their capability or not.
2. There are no security considerations while providing this architecture.

# Chapter IV

# SOFTWARE REQUIREMENT SPECIFICATIONS

## 4.1 Product Overview

For private cloud computing environment, we need to make sure that infinite computing resources are available on demand; however in the current architectures like Eucalyptus, there is a single server or Cluster Controller (CC) which is the sole provider of virtual machines to the users. Despite being an extremely powerful machine it has the constraint of finite resources for launching Virtual Machine (VM) instances. Our model describes a novel cloud approach based on master-slave concept where one master and multiple slaves serve the resources to the clients. This architecture allows new clients to request virtual machines, and the server makes the choice of running the requested virtual machine either on previously available slaves, or on the clients who are recently registered into a set of slaves. This approach provides the functionalities such as launching VM instance on slaves, allows registration and deregistration of client and relocation of VM instances.

### 4.1.1 Hardware Requirements

For developing the architecture the following are the Hardware Requirements:

**Recommended (Master)**

| | |
|---|---|
| Server | : 64 bit Ubuntu 9.10 with VT enabled |
| RAM | : 4 GB DDR2 |

**Recommended (Client)**

| | |
|---|---|
| Server | : 32/64 bit Ubuntu 9.10 with VT enabled |
| RAM | : 2 GB DDR2 |

### 4.1.2 Software requirements

For developing the architecture the following are the Software Requirements:

Operating System: Linux/Ubuntu 9.10 (Karmic)

| | |
|---|---|
| Language | : Python |
| Database | : MySQL |

### 4.1.3 Software Product Feature

**Product perspective**

- The proposed Master Slave architecture a new proposed architecture for the setting up of Virtual Computing Lab. It overcomes the drawbacks of existing open source architecture (bottleneck).It enhances the resources availability by making the requesting node as the slave after it meets it requirements .Here making the node as the slave is the main focus of the project, there is also a algorithm for the scheduler to schedule the requests accordingly. So both aspects are covered in detail within this document.

**Product features**

- The main features of proposed architecture are that the ddrawbacks of existing open source architectures are removed (bottleneck), adaptive cloud approach is carried out, availability of sufficient resources is there and scheduler algorithm has prioritization, critical Limit.

**User characteristics**

- The application is intended to be used by any user who has interest in using the resources available in the existing master-slave architecture.

### 4.1.4 Constraints

1. The participants should not be changing their IP.
2. A client will be allowed to opt for being “not becomes a slave” only two times. After that if he wants to make use of architecture he has to become a slave

3. We also need to take into account that some users will not want to share their resources i.e. not willing to become slave and therefore we need to warn or confirm

with the user that they might not be getting the priority when they are requesting for the resources.

### 4.1.5 Assumptions & dependencies

1. The user has a good intent to use the resources.
2. The users will be willing to enlist themselves as slaves.
3. Users will be making use of the resources for the limited time.

## 4.2 Software System Attributes

### 4.2.1 Reliability

The architecture i.e. the master and the slaves are required to be actively connected to the scheduler node. As long as they are connected and scheduler is working properly the architecture would be perfectly reliable for performing its objective.

### 4.2.2 Availability

1. The architecture shall be available freely and will run in a stable state at all times.
2. The master shall be available and will run in a stable state at all times.
3. Once the scheduler is implemented the architecture will be readily available and will be working automatically.

.

### 4.2.3 Security

1. The system shall require a password to allow a node to logon.
2. The system shall check the IP and the MAC address of the nodes when requesting to connect second time after being enlisted as slave.
3. Only administrator/master will have the database administration rights.

### 4.2.4 Maintainability

1. The various components of the system have well defined and mathematically proven interfaces that do not require any changes. The scheduling approaches

themselves may be changed, while keeping the interface constant, without affecting the functionality of the system.

2. The scheduling table should be up to date with the latest context information queried to the server

### 4.2.5 Portability

Any node willing to make use of the architecture and its resources will be able to meet their requirements from it when approved by the Master node.

### 4.2.6 Performance

Our architecture removes the drawbacks of the existing architectures and performance in terms of CPU, Memory and network utilization will be increased and though there will be load on the system it will not result into bottleneck.

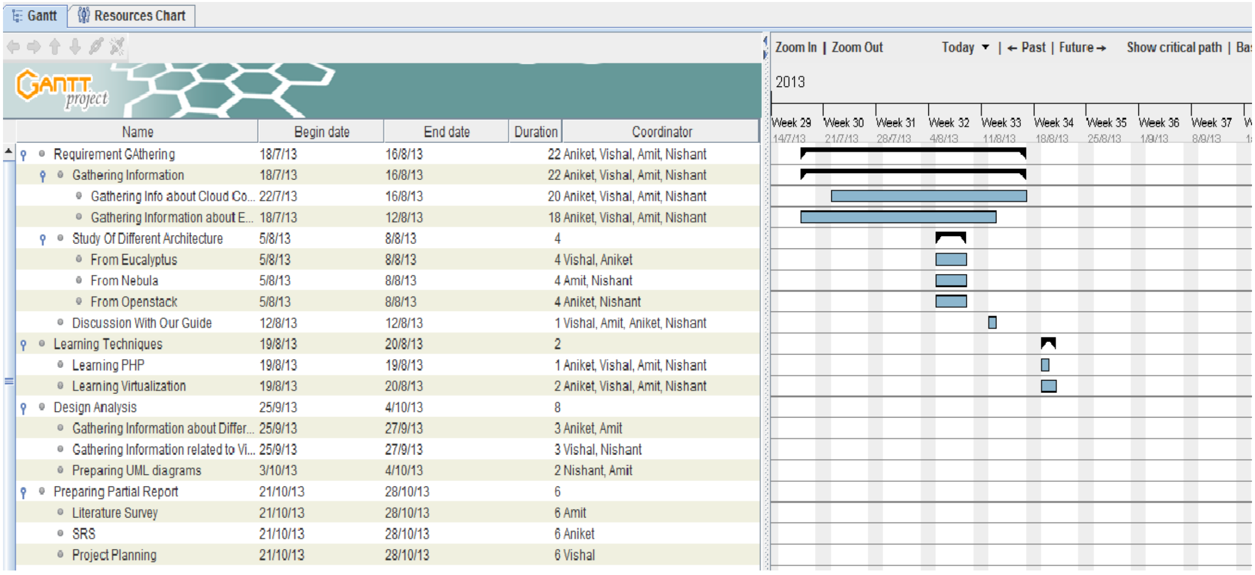
## 4.3 Database Requirements

1. A database is required to store the node number along with their corresponding IP and MAC address on the Master side.
2. Status log and information of the existing users.
3. A database is required to store the nodes with their respective information about the resources available and their status.
4. A table to hold the waiting requests.

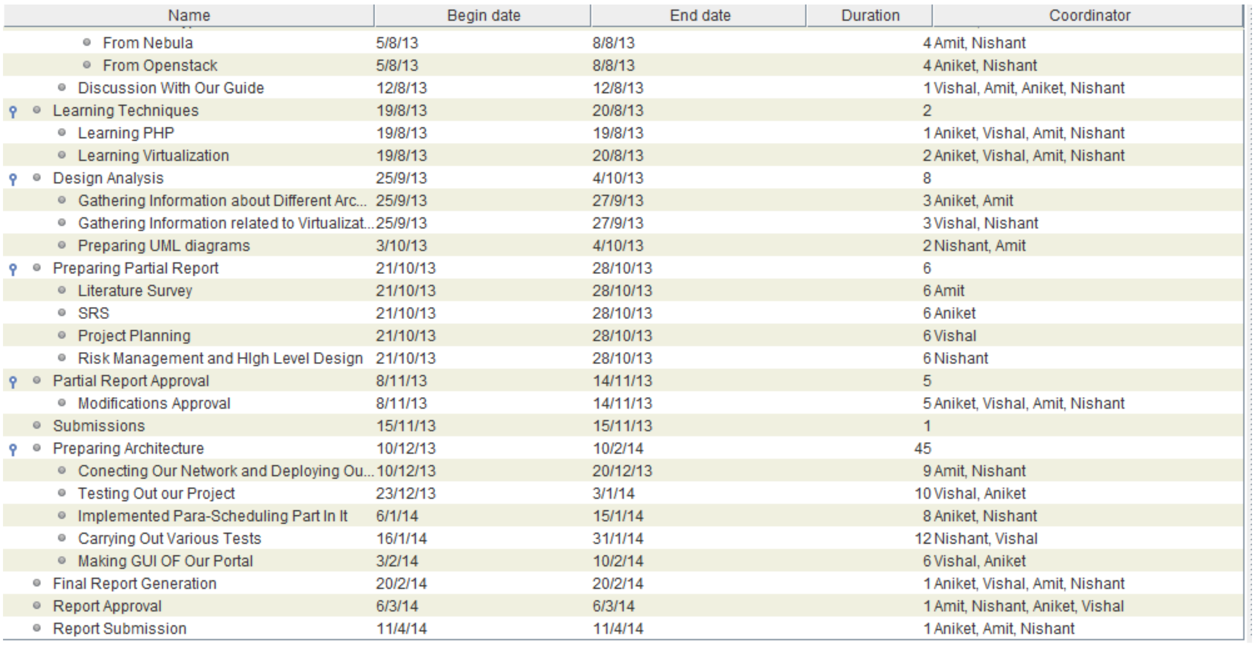
# Chapter V

# SOFTWARE PROJECT PLAN

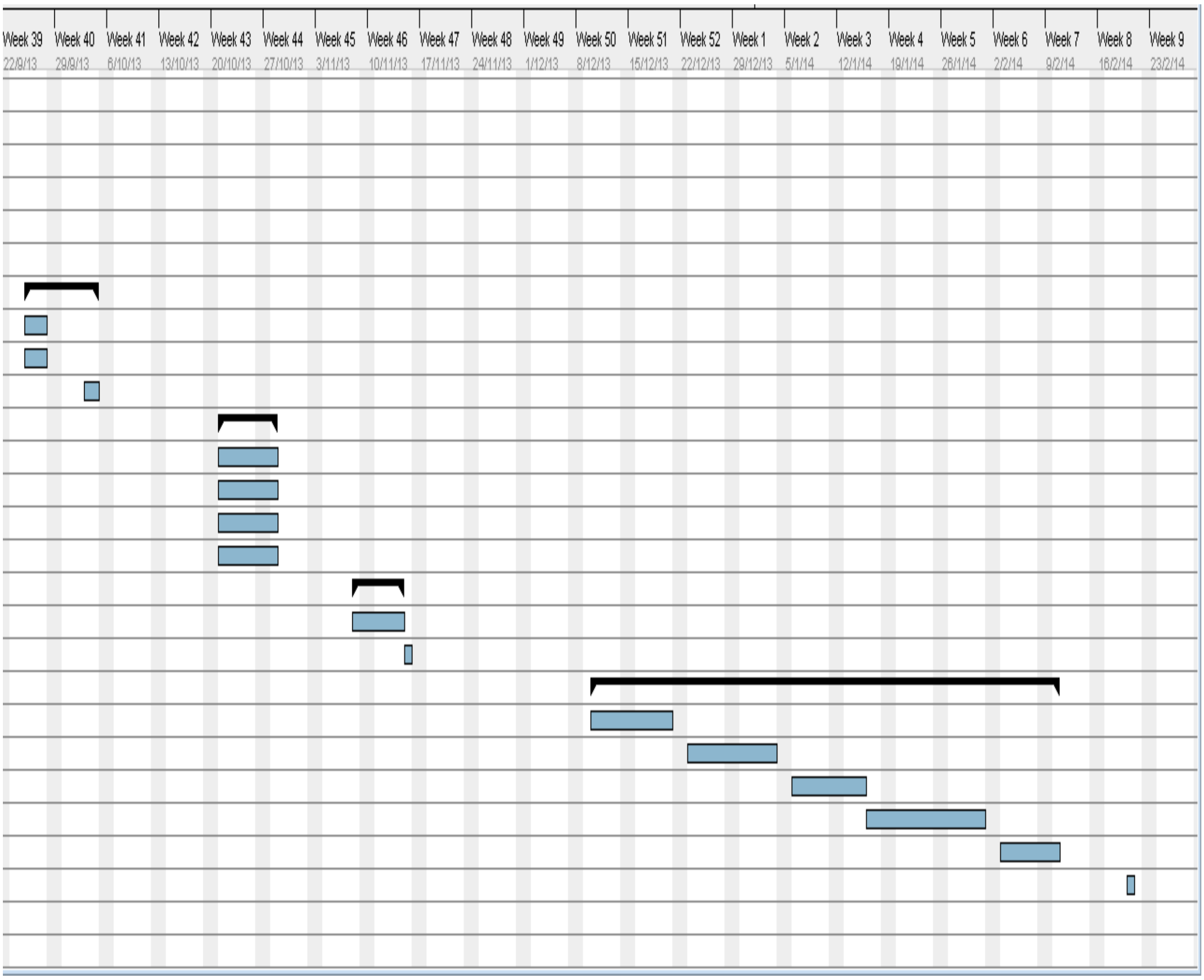
## 5.1 Project Scheduling - Gantt chart



| Name | Begin date | End date | Duration | Coordinator |
|---|---|---|---|---|
| Requirement GAthering | 18/7/13 | 16/8/13 | 22 | Aniket, Vishal, Amit, Nishant |
| Gathering Information | 18/7/13 | 16/8/13 | 22 | Aniket, Vishal, Amit, Nishant |
| Gathering Info about Cloud Co... | 22/7/13 | 16/8/13 | 20 | Aniket, Vishal, Amit, Nishant |
| Gathering Information about E... | 18/7/13 | 12/8/13 | 18 | Aniket, Vishal, Amit, Nishant |
| Study Of Different Architecture | 5/8/13 | 8/8/13 | 4 | |
| From Eucalyptus | 5/8/13 | 8/8/13 | 4 | Vishal, Aniket |
| From Nebula | 5/8/13 | 8/8/13 | 4 | Amit, Nishant |
| From Openstack | 5/8/13 | 8/8/13 | 4 | Aniket, Nishant |
| Discussion With Our Guide | 12/8/13 | 12/8/13 | 1 | Vishal, Amit, Aniket, Nishant |
| Learning Techniques | 19/8/13 | 20/8/13 | 2 | |
| Learning PHP | 19/8/13 | 19/8/13 | 1 | Aniket, Vishal, Amit, Nishant |
| Learning Virtualization | 19/8/13 | 20/8/13 | 2 | Aniket, Vishal, Amit, Nishant |
| Design Analysis | 25/9/13 | 4/10/13 | 8 | |
| Gathering Information about Differ... | 25/9/13 | 27/9/13 | 3 | Aniket, Amit |
| Gathering Information related to Vi... | 25/9/13 | 27/9/13 | 3 | Vishal, Nishant |
| Preparing UML diagrams | 3/10/13 | 4/10/13 | 2 | Nishant, Amit |
| Preparing Partial Report | 21/10/13 | 28/10/13 | 6 | |
| Literature Survey | 21/10/13 | 28/10/13 | 6 | Amit |
| SRS | 21/10/13 | 28/10/13 | 6 | Aniket |
| Project Planning | 21/10/13 | 28/10/13 | 6 | Vishal |



| Name | Begin date | End date | Duration | Coordinator |
|---|---|---|---|---|
| From Nebula | 5/8/13 | 8/8/13 | 4 | Amit, Nishant |
| From Openstack | 5/8/13 | 8/8/13 | 4 | Aniket, Nishant |
| Discussion With Our Guide | 12/8/13 | 12/8/13 | 1 | Vishal, Amit, Aniket, Nishant |
| Learning Techniques | 19/8/13 | 20/8/13 | 2 | |
| Learning PHP | 19/8/13 | 19/8/13 | 1 | Aniket, Vishal, Amit, Nishant |
| Learning Virtualization | 19/8/13 | 20/8/13 | 2 | Aniket, Vishal, Amit, Nishant |
| Design Analysis | 25/9/13 | 4/10/13 | 8 | |
| Gathering Information about Different Arc... | 25/9/13 | 27/9/13 | 3 | Aniket, Amit |
| Gathering Information related to Virtualizat... | 25/9/13 | 27/9/13 | 3 | Vishal, Nishant |
| Preparing UML diagrams | 3/10/13 | 4/10/13 | 2 | Nishant, Amit |
| Preparing Partial Report | 21/10/13 | 28/10/13 | 6 | |
| Literature Survey | 21/10/13 | 28/10/13 | 6 | Amit |
| SRS | 21/10/13 | 28/10/13 | 6 | Aniket |
| Project Planning | 21/10/13 | 28/10/13 | 6 | Vishal |
| Risk Management and HIgh Level Design | 21/10/13 | 28/10/13 | 6 | Nishant |
| Partial Report Approval | 8/11/13 | 14/11/13 | 5 | |
| Modifications Approval | 8/11/13 | 14/11/13 | 5 | Aniket, Vishal, Amit, Nishant |
| Submissions | 15/11/13 | 15/11/13 | 1 | |
| Preparing Architecture | 10/12/13 | 10/2/14 | 45 | |
| Conecting Our Network and Deploying Ou... | 10/12/13 | 20/12/13 | 9 | Amit, Nishant |
| Testing Out our Project | 23/12/13 | 3/1/14 | 10 | Vishal, Aniket |
| Implemented Para-Scheduling Part In It | 6/1/14 | 15/1/14 | 8 | Aniket, Nishant |
| Carrying Out Various Tests | 16/1/14 | 31/1/14 | 12 | Nishant, Vishal |
| Making GUI OF Our Portal | 3/2/14 | 10/2/14 | 6 | Vishal, Aniket |
| Final Report Generation | 20/2/14 | 20/2/14 | 1 | Aniket, Vishal, Amit, Nishant |
| Report Approval | 6/3/14 | 6/3/14 | 1 | Amit, Nishant, Aniket, Vishal |
| Report Submission | 11/4/14 | 11/4/14 | 1 | Aniket, Amit, Nishant |



## 5.2 Risk Management

Risk management is concerned with identifying risks and drawing up plans to minimize their effect on a project.

A risk is a probability that some adverse circumstance will occur.

- Project risks affect schedule or resources.

- Performance risks affect the quality or performance of the software being developed.

- Technology risks affect the technical aspects of the project. For example database, function reusability

The main areas of the project that involve significant risks are:

**Performance Risks:**

- The architecture is bound to not work properly if there are change in IP address of any node.

- The number of hops taken by a request travelling from a client machine towards the master machine if are increased can result into packet loss.

**Project Risks:**

- The project may have security breaches as it is cloud architecture.
- Majorly the risk is associated with networking and IP part.So, care should be taken that the IP's used are from a free pool and are statically configured.

**Technical Risks:**

- Due to network congestion there may be a case that we lose some of the files while information is getting transferred.
- If a node is not freeing up the resources it is using

## 5.3RMMM (Risk Mitigation, Monitoring and Management) Plan

1] **Risk:** Insufficient system environment

- **Mitigation:** The application is to be run in virtualized environment on multi computers. The Master should enlist all its available resources systematically. In case of resource management inadequacy to support the environment may not be generated, leading to product failure.
- **Monitoring:** The master should be aware of current available resources in synchronized manner.
- **Management:** In case of insufficient resources the admin should diagnose the system.

2) **Risk:** Deviation from software engineering standards

- **Mitigation:** While possible deviation from possible software engineering standards is unlikely to occur all team members have a full understanding of the architecture and how we plan to implement the architecture.

- **Monitoring:** Technical reviews involving comparison between proposed architecture in documentation and the actual architecture implemented and comparing various experiment test results [Network, CPU and Memory utilization] with expected results. This will help to determine if deviation will occur. All the relevant documents must be as complete and accurate as possible to ensure the work will conform to express software engineering standards.
- **Management:** Should deviation occur, steps must be taken to guide the project back within the Standards expressed in the accompanying documents. Technical reviews help to determine what must be done to keep the project the inline with established software engineering standards.

# Chapter VI

# DIAGRAMS AND DATABASE DESIGN

## 6.1 Block /Architecture Diagram

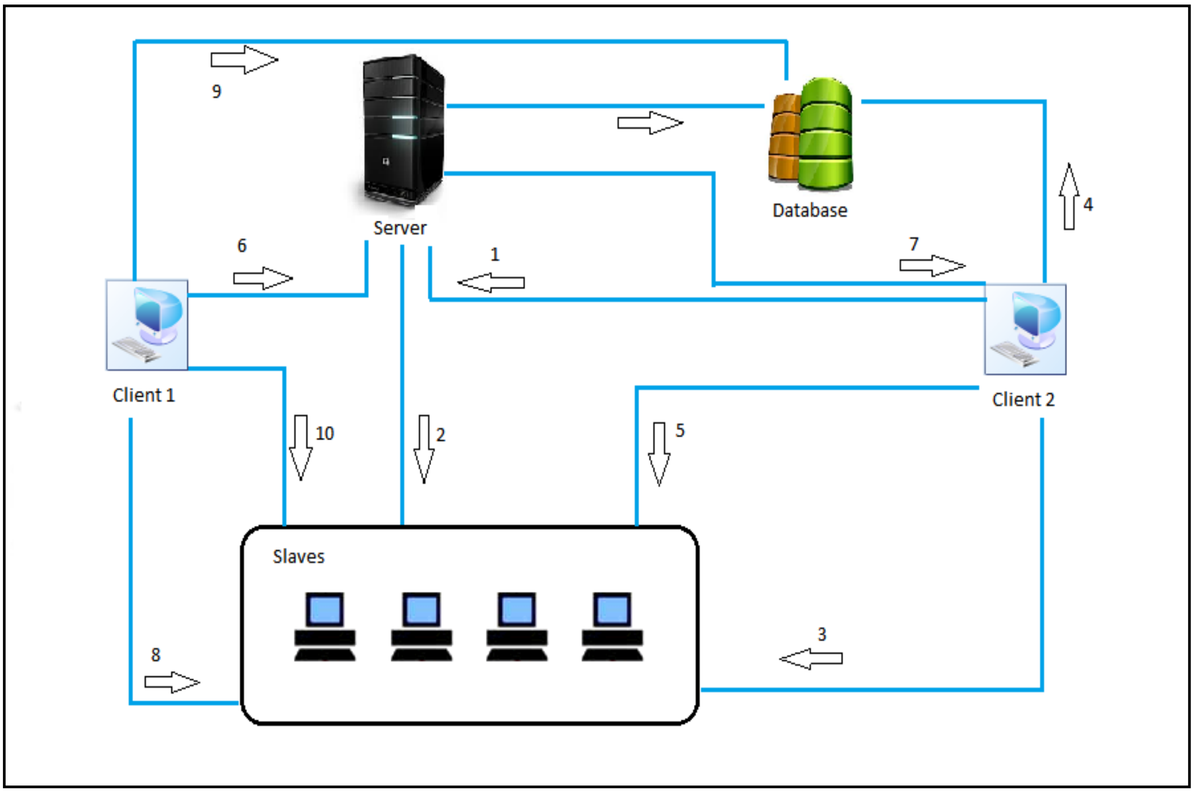


**Figure 11. Flow of events**

**Flow of events:**

**The description of flow of events is as follows:**

1. Client 1 requests an instance to the Master.

2. This request is evaluated by the master machine and accordingly forwards this request to the suitable slave machine.

3. The Master machine gives the IP address of slave machine to the client 1 for any further connections. Once the client 1 knows the IP address of the slave, it directly connects with the slave machine.

4. Client 1 sends its address to the database so that it can perform the function of slave for any future calls to the master.

5. Client 1 is added to the pool of resources, and is registered as a slave with the master.

6. New Client 2 sends request to the Master.

7. Master forwards request to client 1 provided that sufficient resources are available with client1.

8. If Client 1 does not have enough resources, the master forwards the request to the set of slaves.

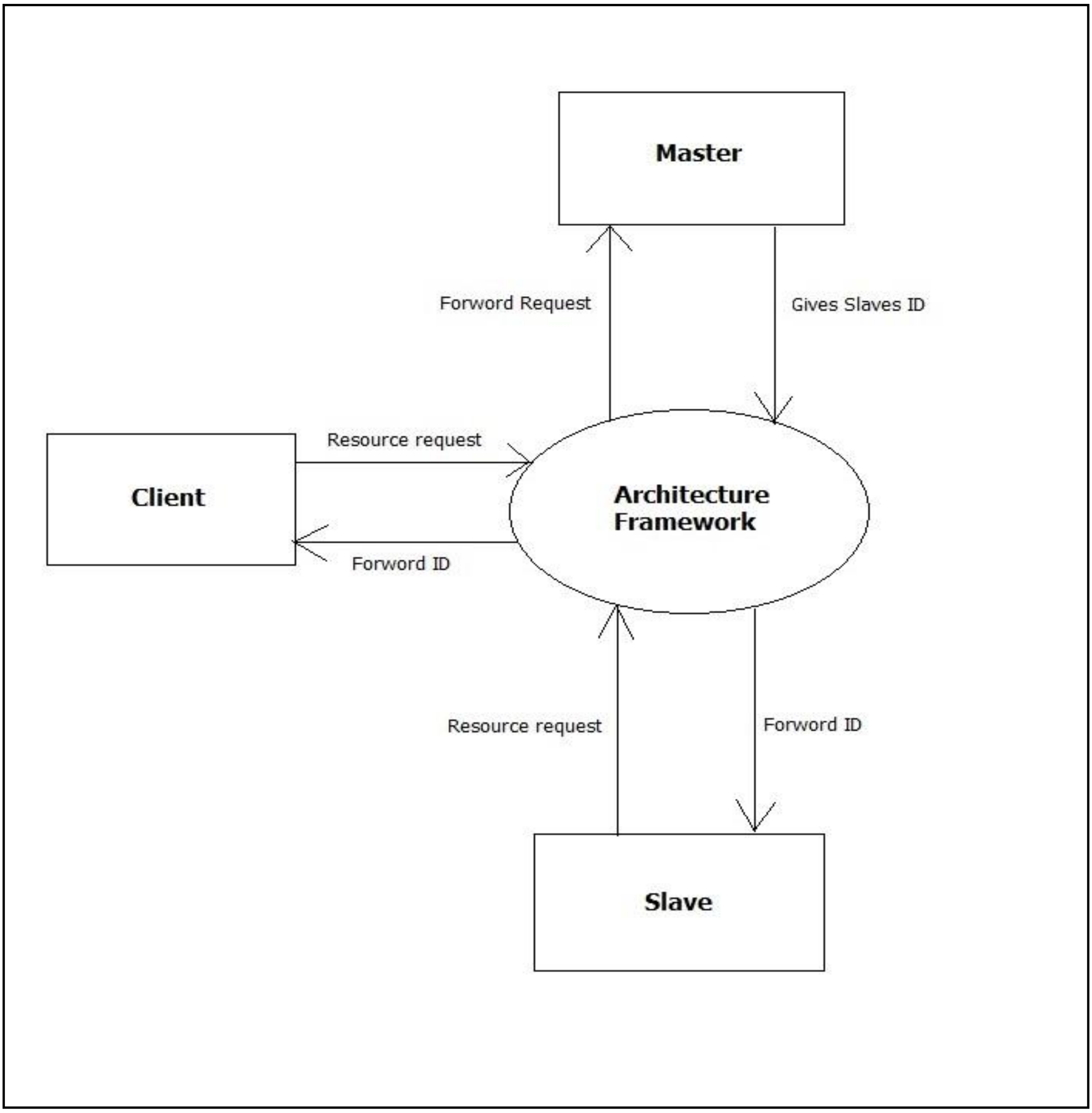
9. Client 2 sends its address to the database so that it can perform the function of slave for any future calls to the master.

10. Client 2 is added to the pool of resources, and is registered as a slave with the master.

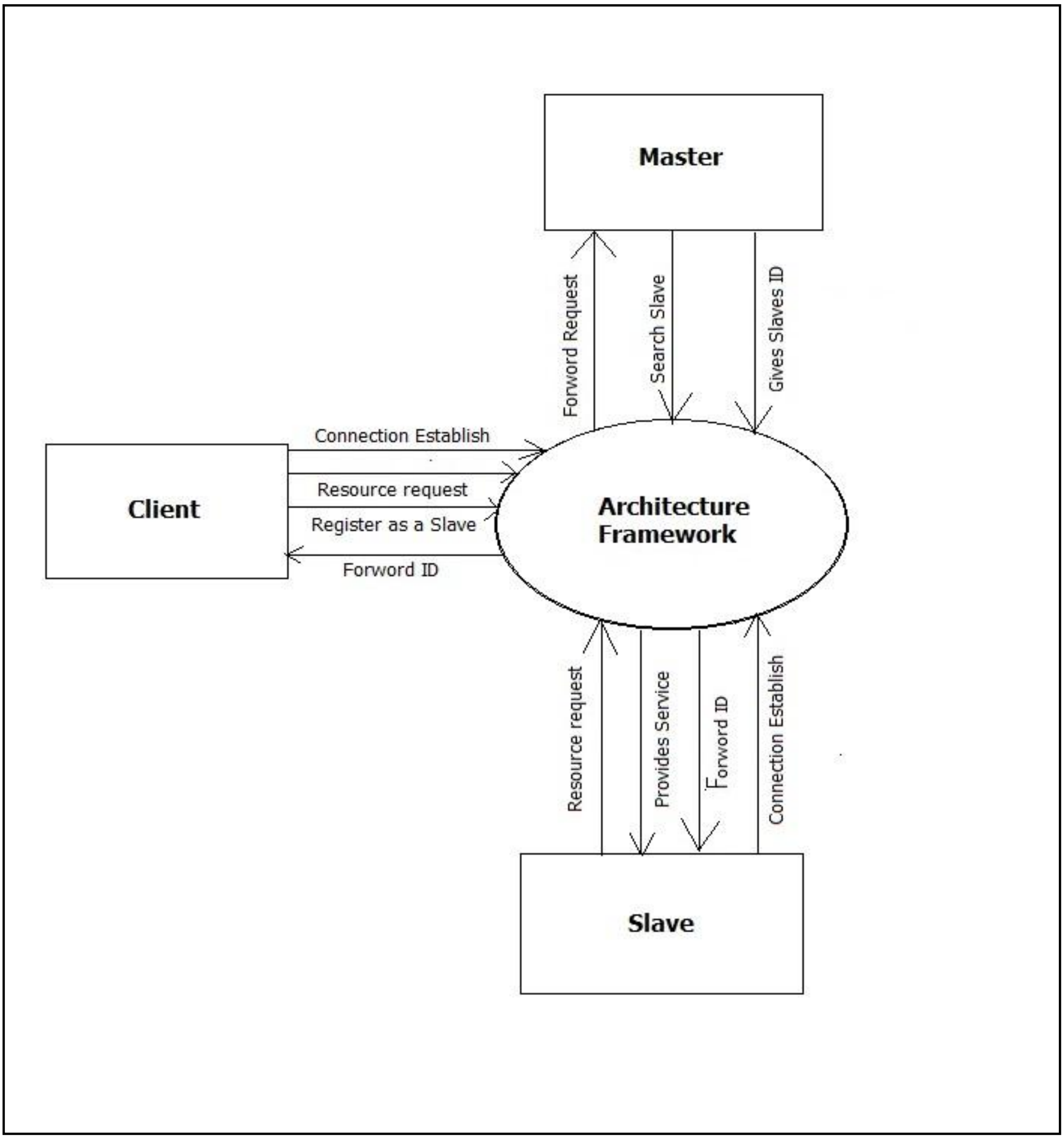
Thus, this architecture ensures that there are always sufficient resources available at the disposal of the master, and thus can accept any number of incoming connections.

## 6.2 Data flow diagram

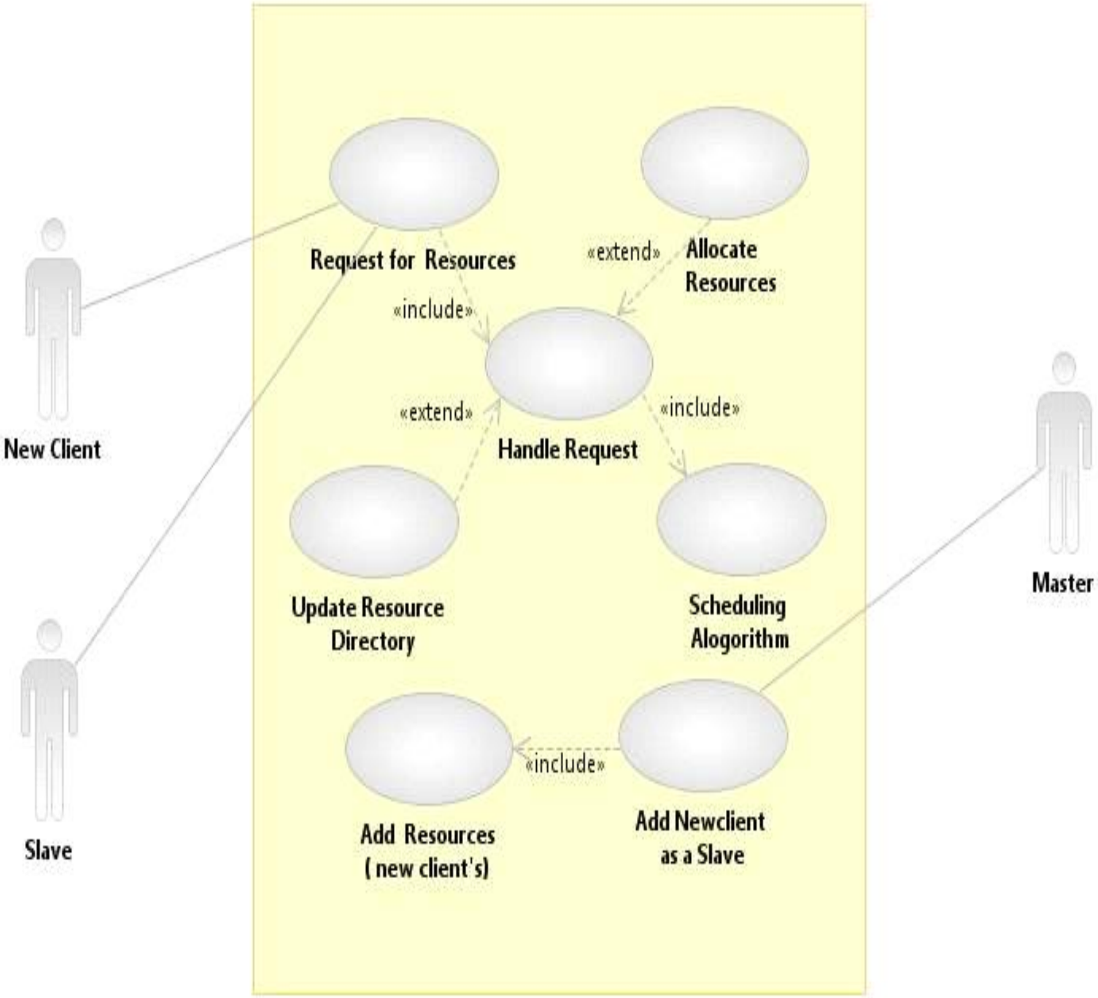
### Level 0:



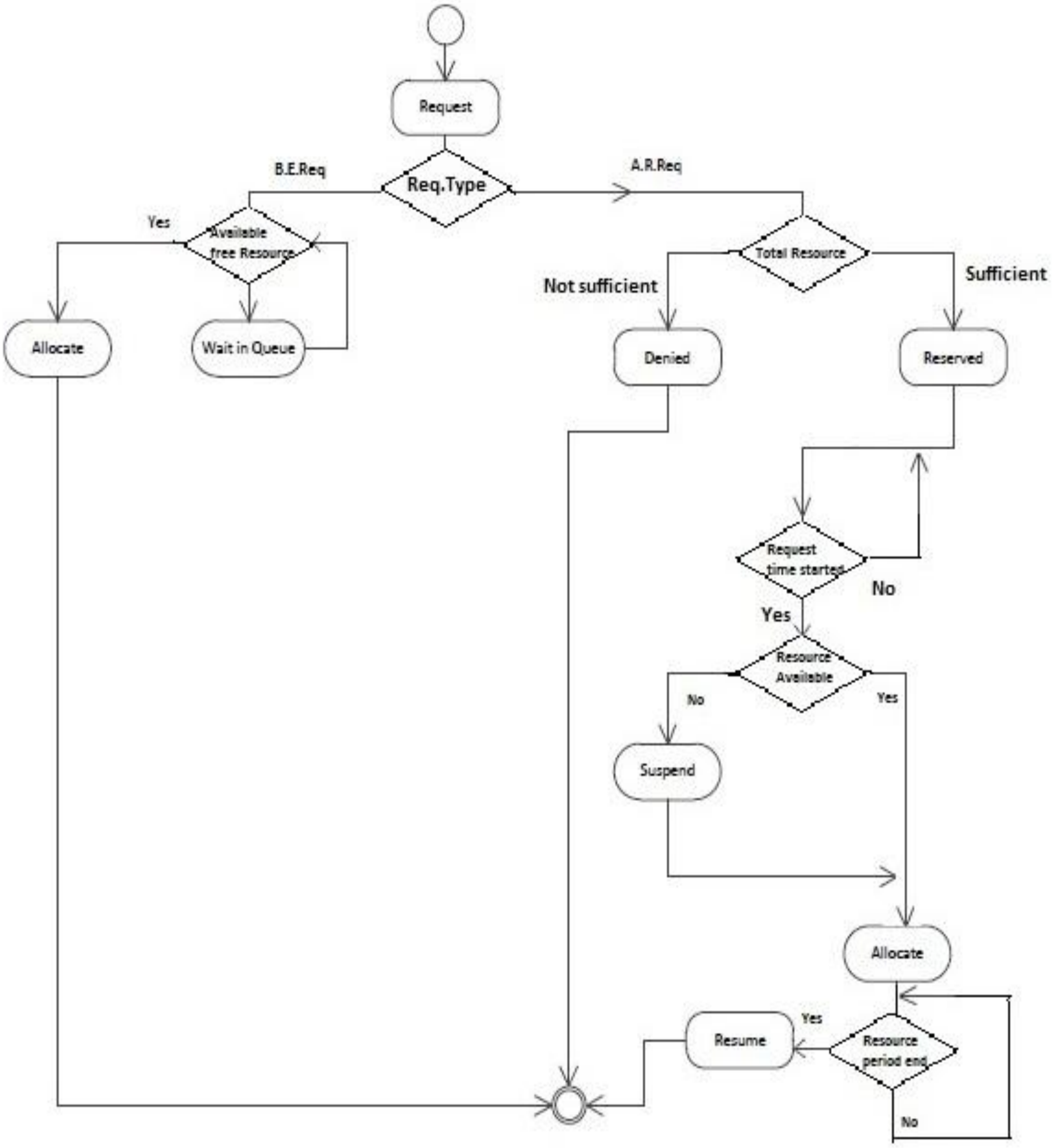
**Level 1:**


Master
Forword Request
Search Slave
Gives Slaves ID
Client
Connection Establish
Resource request
Register as a Slave
Forword ID
Architecture Framework
Resource request
Provides Service
Forword ID
Connection Establish
Slave

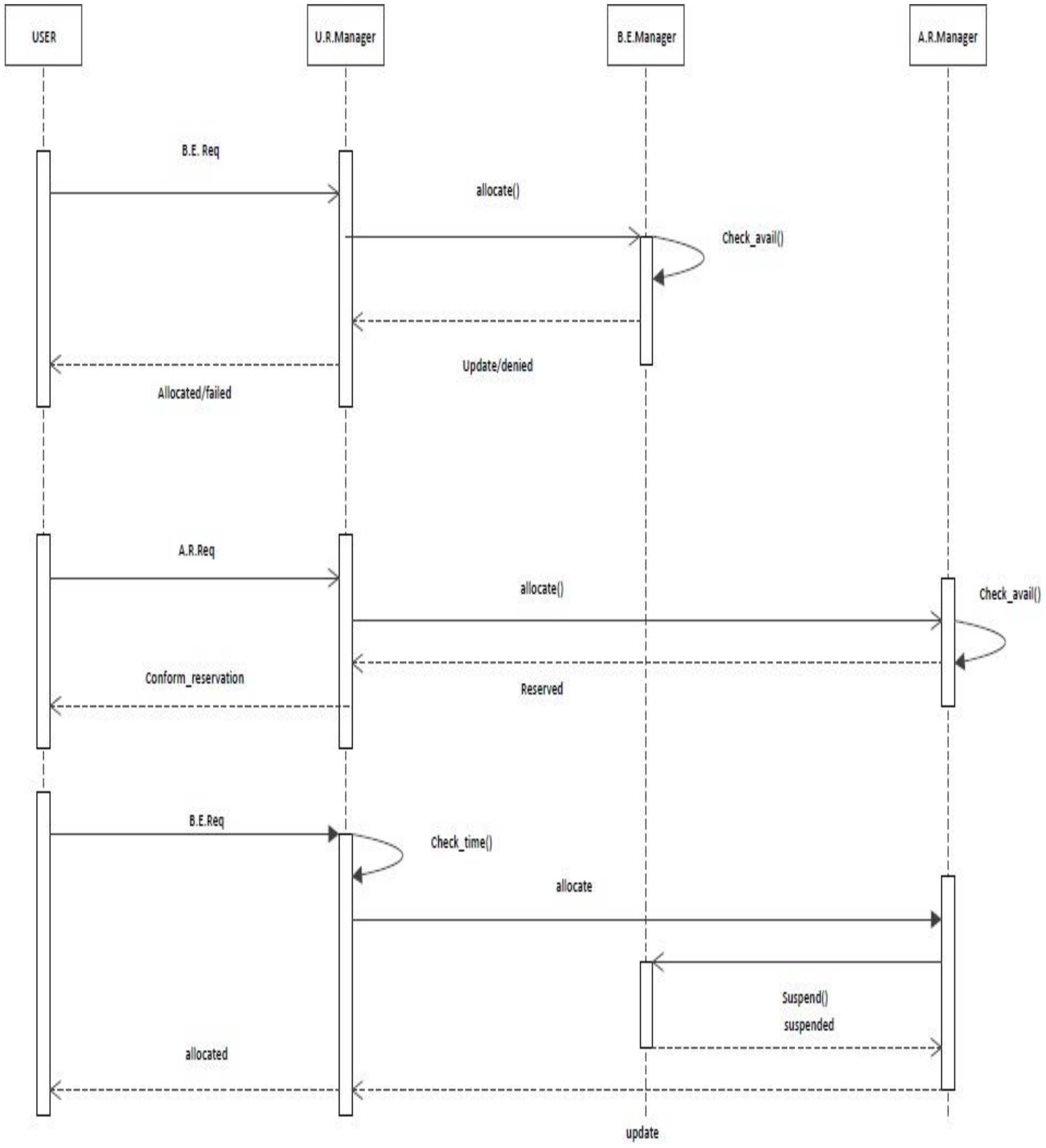
## 6.3 Use Case Diagram



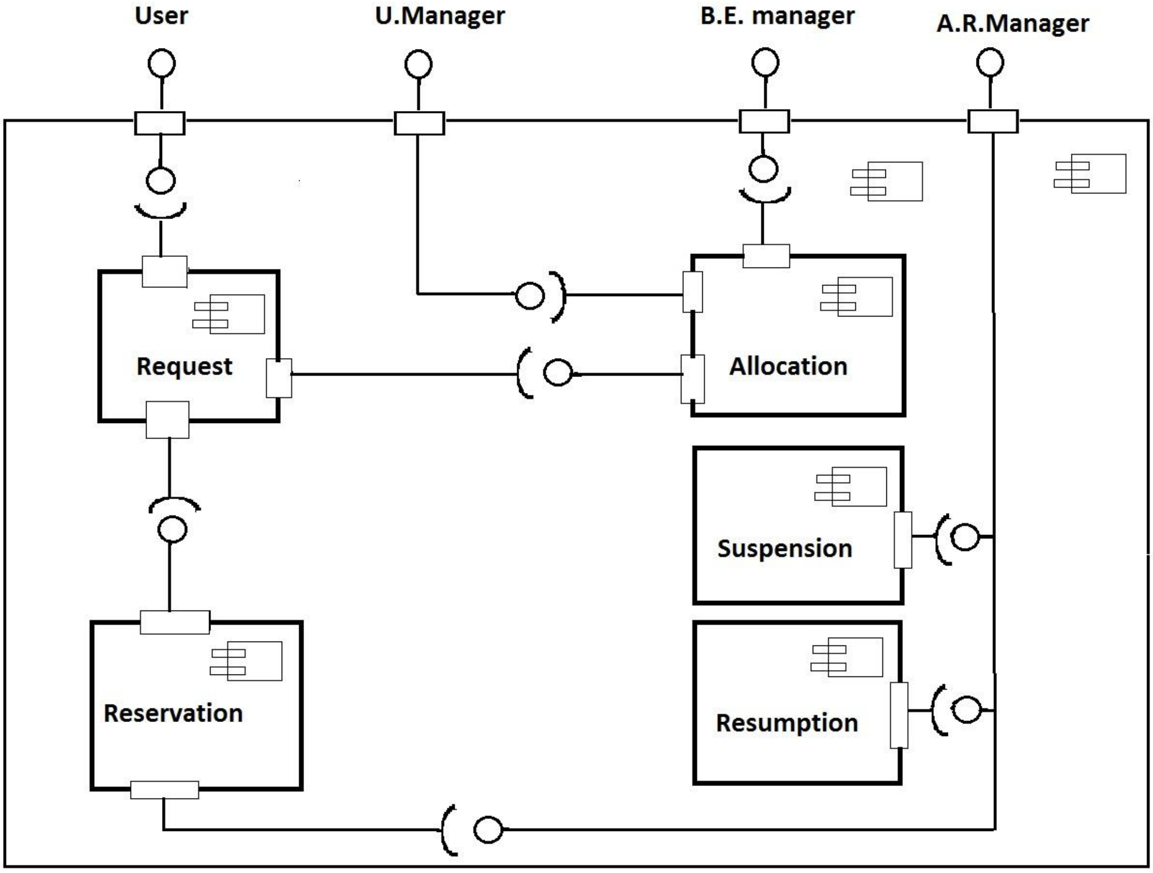
## 6.4 Activity diagram:



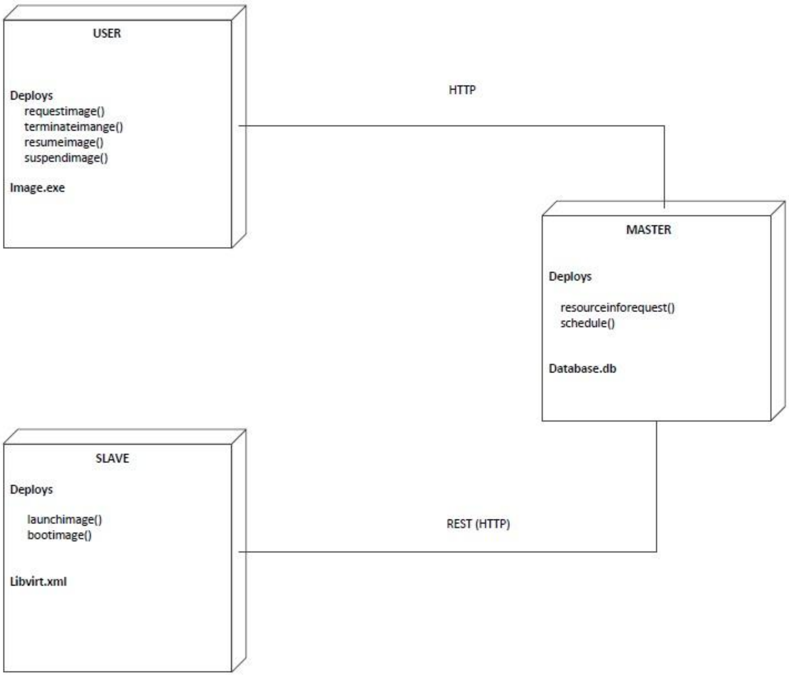
## 6.5 Sequence diagram:



## 6.6 Component diagram:



## 6.7 Deployment diagram:



## 6.8 Database design: Tables:

*Slave table* :

| Attribute | Attribute value : Example for one entry |
|---|---|
| Name | SLAVE_88 |
| Password | 13432423432 |
| Ip_address | 10.10.9.233 |
| Total_ram | 343534 |
| Total_cores | 12 |
| Total_hdd_size | 545454 |
| Used_ram | 2323 |
| Used_cores | 2 |
| Used_hdd_size | 43453 |
| Status | OK |

*Kernel table* :

| Attribute | Attribute value : Example for one entry |
|---|---|
| Id | EKI-1234567 |
| Name | EKI-1234567 |
| Description | EKI-1234567 |

*Groups table* :

| Attribute | Attribute value : Example for one entry |
|---|---|
| Group_name | CL II |
| Group_owner | CL II |
| Group_Desription | CL II |

*Ram Disk table* :

| Attribute | Attribute value : Example for one entry |
|---|---|
| Id | ERI-1234567 |
| Name | ERI-1234567 |
| Description | ERI-1234567 |

*Users table* :

| Attribute | Attribute value : Example for one entry |
|---|---|
| User_id | A |
| User_password | A |
| Group_group_name | A |

*Disk Images table* :

| Attribute | Attribute value : Example for one entry |
|---|---|
| Id | EMI-1234567 |

| | |
|---|---|
| Name | EMI-1234567 |
| Description | EMI-1234567 |
| Default_kernel_id | EKI-1234567 |
| Default_ramdisk_id | ERI-1234567 |
| Default_ram | 402445 |
| Default_cores | 2 |
| Default_hdd_size | 204800 |

*Instance table* :

| **Attribute** | **Attribute value : Example for one entry** |
|---|---|
| Id | I-UBUNTU-9AF |
| Disk_id | EMI-1234567 |
| Kernel_id | EKI-1234567 |
| Ramdisk_id | ERI-1234567 |
| Slave_name | SLAVE_73 |
| Ram | 402445 |
| Cores | 2 |
| Hdd_size | 204800 |
| Status | NOTOK |
| Vmtype | BE |
| User | B |

*Host table* :

| **Attribute** | **Attribute value : Example for one entry** |
|---|---|
| Id | 1 |
| Instance_id | I-UBUNTU-B8E |
| Mac_address | 00:16:3E:62:24:14 |
| Private_ip | 10.1.44.236 |
| Public_ip | |

*Reservation_status table* :

| **Attribute** | **Attribute value : Example for one entry** |
|---|---|
| Id | 1 |
| Total_cores | 12 |
| Used_cores | 0 |

# Chapter VII

# INSTALLATION

## 7.1 Installation Setup

Very first recommendation is to go through first installing Eucalyptus as a learning step. Because, installing cloud using eucalyptus would make you familiar with the steps used below. Our Cloud setups consist of master – slave concept. Thus, we have setup cloud using one master machine and one slave.

Also, before you go through the installation steps. Please go roughly with the section "Problems faced during installation and how to solve them" first.

**Machine Configuration**:

**For Master and Slave:**

1. Ubuntu 9.10 Operating system

As it is old version of Ubuntu, it can only be installed on HP machine.

2. 32-bit i3 processor
3. Virtualization Support i.e. machine should be VT enabled ( It can be checked in BIOS settings)
4. RAM 2 GB , HDD 350 GB

**Configuring steps:**

First and most login through root by using command:

```
 master@master  :  su
or
master@master :  sudo bash
```

## 7.2 Static IP Configuration and NTP:

On command prompt edit “interfaces “file in master as well as slave:

**<u>On master:</u>**

```
master@master :   cd /etc/network
master@master :  sudo gedit interfaces
```

**than it contains following information :**

```
auto lo
iface lo inet loopback
```

**Now, add some more information to it, so the file contain should look like as follows ( the addresses may change as per your machines)**

```
auto lo
iface lo inet loopback
auto   eth0
iface   eth0   inet   static
address 10.10.9.235
netmask  255.255.248.0
gateway 10.10.1.250
```

**Similarly, do the “interfaces” file on slave should look like below :**

```
auto lo
iface lo inet loopback
auto   eth0
iface   eth0   inet   static
address 10.10.9.234
netmask  255.255.248.0
gateway 10.10.1.250
bridge_ports eth0
```

**After making the above changes on master as well on slave machine, run the following command to restart the networking:**

```
root@master: /etc/init.d/networking restart
```

The time on all components of the setup will have to be in synchronized .So, install ntp package on master and slave as

```
root@master :# apt-get install ntp
```

**Open the file ntp.conf and the following two lines to make sure the master serves time even when its connectivity to the internet is down. The following settings make sure that the ntp server uses its own clock as the clock source.**

**On master:**

```
Server 127.127.1.1.0
fudge 127.127.1.0 stratum 10
```

**On slave**, give the ip of the master in ntp.conf file as :

```
Server 10.1.46.66
```

**Restart ntp service to make the changes effective on master and slave as**

**root@master : /etc/init.d/ntp restart**

**PRE-PACKAGE INSTALLATION STEP:**

Edit the sources. list file in master and slave by making the following changes :

- Replace “in.archive” by “old-releases “
- Replace “security.ubuntu.com” by “old-releases.ubuntu.com”

Make sure that the system is in complete updated system by following command “

**root@master : sudo apt-get update**

**NOTE: It is must that the NTP and NETWORKING should work .If NTP is not working then further steps would be of no use.**
**So, make sure that NTP is established.**

## 7.3 Installing Cloud Setup

Installing cloud follows a comprehensive list of dependencies that must be satisfied before building cloud or running it. Distribution specific installation instructions provide help that satisfy these dependencies. This list should be useful if cloud is being installed or built on an unsupported distribution.

Prerequisites for compiling from source:

-C compilers
-Psutil files
-Netifaces files
-Sqlite files
Wsgi files
Apache ant 1.6.5 or above
-libc development files
-libvirt development files
-openssl development files
-python
-python development files
To run virtual machine on slaves we need –
Vncviewer
The necessary python libs are :
-Simplejson
-ConfigObj
-Sqlalchemy
-Elixir

One physical machine can play the role of the front-end and the node. User must be root to install and start master-slave components (by default they will run under a different user after start).All commands need to be executed as root.

For Ubuntu 9.10 , run the command to install the required dependency packages:

**root@cloud:~#** apt-get install bzr gcc make apache2-threaded-dev ant openjdk-6-jdk\libvirt-dev libcurl4-openssl-dev dhcp3-server vblade apache2 unzip curl vlan\bridge-utils libvirt-bin kvm vtun

The remaining packages can be installed by the following commands:

**root@cloud:~#** apt-get install python
**root@cloud:~#** apt-get install vncviewer
**root@cloud:~#** apt-get install python-setuptools
**root@cloud:~#** easy_install configobj
**root@cloud:~#** easy_install simplejson
**root@cloud:~#** easy_install elixir
**root@cloud:~#** easy_install sqlite
**root@cloud:~#** easy_install sqlitebrowser
**root@cloud:~#** easy_install sqliteman
**root@cloud:~#** easy_install sqlalchemy
**root@cloud:~#** aptitude install kvm libvirt-bin ubuntu-vm-builder bridge-utils
**root@cloud:~#** apt-get install libvirt-bin
**root@cloud:~#** apt-get install libvirt-doc

### 7.4 STEPWISE INSTALLATION FOR CLOUD (SUMMARIZED)

Below is the stepwise procedure to install the cloud setup. Steps are different for master as well as the slave machine.

## Step 1:

After configuring the machine (as required in above steps ).It is preferable to do “**ssh**” key exchange and **ntp** between two machines. ( for this refer to the section “ Problems faced during installation and how to solve them”)

Initial steps are “static  ip configuration “  and “ pre package installation step” and then continue with the following steps :( or you can configure the ip statically after the installation of packages , but pre-installation step if must .

Go the path:

**On master and slave both: ( PRE-PACKAGE INSTALLATION STEP )**

At the below path , “sources.list” file is present

**root@cloud:~# cd  /etc/apt**

Edit the sources.list file in master and slave by making the following changes:

- Replace “in.archive” by “old-releases “
- Replace “security.ubuntu.com” by “old-releases.ubuntu.com”

Make sure that the system is in complete updated system by following command “

**root@master : sudo  apt-get   update**

Execute the command:

**root@cloud:~# apt-get update**

## Step 2:

**On master and slave both:**

**root@cloud:~#** apt-get install python

**root@cloud:~#** apt-get install python-libvirt

**root@cloud:~#** apt-get install openssh-server

**On Slave and Client :**

**root@cloud:~#** apt-get install libvirt-bin

**root@cloud:~#** apt-get install libvirt-doc

**On master and slave both:**

**root@cloud:~#** apt-get install python-setuptools

**root@cloud:~#** easy_install configobj

**root@cloud:~#** easy_install sqlalchemy

**root@cloud:~#**  easy_install elixir

**root@cloud:~#** easy_install simplejson

**On master :**

**root@cloud:~#** apt-get install sqlitebrowser

## Step 3:

On master and slave copy the “workspace” named folder in the “root” folder as it is.

## Step 4:

To run the code, through command prompt, go the path :

**On master : Path is :** workspace/Cloud43/src/master

**On Slave : Path is :** workspace/Cloud43/src/slave

**For Database access on master:Path is:** workspace/Cloud43/src/master/db

**Running the Code on master:**

**root@cloud/workspace/Cloud43/src/master:~#** python main.py

**Running the Code on Slave:**

**root@cloud/workspace/Cloud43/src/slave:~#** python main.py

## Step 5:

After running the code on both machines, it will show nothing at the command prompt.

On slave machine, in browser enter the following path :

**http://ip address of master:8000/**

## Step 6:

It would show the following windows .Further stepwise procedure is shown in the section of “Result/Analysis/Discussion”. This is because the further procedure goes through snapshots of various steps.
Also, some steps consists of executing certain command. For that , you can refer to the 9th point in section “ Problems faced during installation and how to solve them”.

## 7.5 Problems faced during installation and how to solve them

1] Usually, the basic problem is of root login. When we log on using “su” command and enter the password, it says “Authentication failure” .At this time, you can using following command .

**root@master : sudo bash**

and then enter the password

Or

**root@master : sudo passwd**

then you can change the root password.

2] Mostly the problem comes of network. As we make changes in “interfaces” file. Everytime when we restart the machine, the internet connection gets disconnected. This is because we edit the networking file. To solve this problem:

Edit the interfaces file from the /etc/network

Cut or delete the lines that we appended before.

Run the commands:

**root@master : /etc/init.d/networking restart**

**root@master :gksu service network-manager restart**

By doing this the internet connection is established again.

At this time, as we need to work the static ip ( the interfaces file) .

Again edit the file and append the lines.

Now, run only below commands:

```
root@master: /etc/init.d/networking restart
```

```
root@master: /etc/init.d/ntp restart
```

But not the “gksu” command.  And then you can continue with further steps. This issue needs to be addressed on every **RESTART** of the machine. Also, it should be done on both machines.

**NOTE: Also, Please ensure that you take a free IP range from the authority or the provider , before the startup of the  installation.**

3] When we run the code i.e “main.py “on command prompt. It gives the error related to “address space already in use “. This is because; we try to run the code again on the same command prompt. Remember, the command **“python main.py”** would be executed only once in a command prompt. If you need to again the code, preferably use another command prompt.
Also , you can use following commands at this issues :

```
root@master /workspace/Cloud43/src/master: pkill python
```

```
root@master /workspace/Cloud43/src/master: pkill main.py
```

Same is applicable for slave machine.

4] One of the initial steps before going through is to “ssh” key exchange between master and slave. It should be verified before carrying the installation of cloud.
<u>Steps for ssh key exchange are as follows :</u>
You want to use Linux and OpenSSH to automize your tasks. Therefore you need an automatic login from host A / user a to Host B / user b. You don't want to enter any passwords, because you want to call ssh from a within a shell script.

First log in on A as user a and generate a pair of authentication keys. Do not enter a passphrase:

```
a@A:~> ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/a/.ssh/id_rsa):
Created directory '/home/a/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/a/.ssh/id_rsa.
Your public key has been saved in /home/a/.ssh/id_rsa.pub.
The key fingerprint is:
3e:4f:05:79:3a:9f:96:7c:3b:ad:e9:58:37:bc:37:e4 a@A
```

Now use ssh to create a directory ~/.ssh as user b on B. (The directory may already exist, which is fine):

```
a@A:~> ssh b@B mkdir -p .ssh
b@B's password:
```

Finally append a's new public key to b@B:.ssh/authorized_keys and enter b's password one last time:

```
a@A:~> cat .ssh/id_rsa.pub | ssh b@B 'cat >> .ssh/authorized_keys'
b@B's password:
```

From now on you can log into B as b from A as a without password:

```
a@A:~> ssh b@B hostname
B
```

Note : Depending on your version of SSH you might also have to do the following changes:

- Put the public key in .ssh/authorized_keys2
- Change the permissions of .ssh to 700
- Change the permissions of .ssh/authorized_keys2 to 640

5] Ntp is one the important issue which needs to be solved before installation of cloud. This is because, we need to synchronize both the machines with respect to time.

**Restart ntp service to make the changes effective on master and slave as**

**root@master : /etc/init.d/ntp restart**

6] Take the unique or free ip address range from the authority and assign it to the machines.

7] To access the **“root”** folder there are two methods:

- First Way
  - Press “ Alt+F2”.
  - Type “ gksu nautilus”.
  - Click on “run”.
  - Enter password.
- Second Way
  - On command prompt you can use a command to change the permissions of the folder which you need to access:

    **root@cloud:~#** chmod 755 /root –R

8] To access the database:

**root@master /workspace/Cloud43/src/master/db :** sqlitebrowser   cloud43.db

9] Commands used:

To check list of instances running : **root@slave :** virsh list  
To launch vncviewer : **root@slave :** vncviewer   0.0.0.0  
To disable  firewall : sudo ufw disable  
To check the status of firewall : ufw status  
To flush the iptables : sudo iptables   -F  
To destroy the instance : virsh destroy  name_of_image

10] What to do after the VM is launch on the slave ?

After launching the VM on to the slave machine.This instance of VM should be accessible through a client machine now.To do this , from client machine's command prompt , you can use the following command to access :

Note: install vncviewer first

```
root@client :vncviewer    ip_address_VM_ip  : 5091
```

11] Among one of the initial changes is to escape from a error which usually comes at the first time while running the code .

The error is about “File "/usr/local/lib/python2.6/dist-packages/Elixir-0.7.1-py2.6.egg/elixir/entity.py", line 17, in <module>ImportError: cannot import name ScopedSession”

To solve this , do the following changes :

**Go to the path mentioned in the above error and edit the “entity.py” file as follows :**

Orginal  file contains looks as follows :

```
from sqlalchemy.orm import MapperExtension, mapper, object_session, \
                    EXT_CONTINUE, polymorphic_union, ScopedSession, \
                    ColumnProperty
```

Change file contain and make it as below :

```
from sqlalchemy.orm import MapperExtension, mapper, object_session, \
                    EXT_CONTINUE, polymorphic_union,scoped_session as ScopedSession, \
                    ColumnProperty
```

# Chapter VIII
# IMPLEMENTATION

## 8.1 Master's Code

### File "main.py"

```
'''
Main script which runs on the Master
'''
import socket
import math
from twisted.web.resource import Resource
from twisted.web.server import Site
from twisted.internet import reactor
import simplejson as json
import urllib
import urllib2
from configobj import ConfigObj
from models import *
from sqlalchemy import desc
from twisted.web import static
import datetime
import time
from time import strftime,strptime
from sqlalchemy.processors import str_to_date
from sqlalchemy.sql.expression import or_
from RegisterSlave import *

#print DateTime.timezone
global reserv_flag
reserv_flag = 0
global reserv_id
reserv_id=-1
user = ""
#Add self to classpath
paths = ['.', '..']
for path in paths:
	if path not in sys.path:
		sys.path.append(path)

#Reset all slaves to down.
hosts = Host.query.all()
for host in hosts:
	session.delete(host)
	session.flush()
session.commit()

class RootResource(Resource):

	def __init__(self, config):
		Resource.__init__(self)
		self.putChild("addImage", AddImageResource())
```

```
				self.putChild("login", LoginForm())
				self.putChild("reservation", ReservationForm())
				self.putChild("group", GroupForm())
				self.putChild("user", UserForm())
				self.putChild("slaves", SlavesResource())
				self.putChild("slave", SlaveRootResource())
				self.putChild("resume", ResumeForm())
				self.putChild("intermed", Inter())
				self.putChild("image", DiskImageRootResource(config["dhcp_server"],
config["host_factory"]))
				self.putChild("instance", InstanceRootResource())
				self.putChild("launchimage", LaunchImageForm())
				#self.putChild("intermed", Inter())
				self.putChild("index",IndexForm())
				self.putChild("register",RegisterslaveForm())
				self.putChild("registers",Registers())
				self.putChild("admin",AdminLoginForm())
				self.putChild("checkadmin", CheckAdminLogin())

		def getChild (self, path, request):
				if path=="":
						return IndexForm()


##################################################


class AdminLoginForm(Resource):
		def __init__(self):
				Resource.__init__(self)

		def render_GET(self, request):
				return """ <HEAD>
  <style type="text/css">
		.rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
		.mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
		.home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>
<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
				<H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
				</FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
			<img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
  <BR><BR><br>
  <FORM ACTION="/checkadmin/">
        <H1><CENTER><U><I><B><FONT SIZE="" COLOR="#996633">Admin
Login</FONT></B></I></U></CENTER></H1>

        <CENTER>
        <TABLE CELLSPACING="25">
        <TR>
                <TD><FONT SIZE="5" COLOR="GREEN">Admin ID</FONT></TD>
                <TD><FONT SIZE="5" COLOR="GREEN">:-</FONT></TD>
                <TD><INPUT TYPE="text" NAME="uid"></TD>
        </TR>

        <TR>
                <TD><FONT SIZE="5" COLOR="GREEN">Password</FONT></TD>
                <TD><FONT SIZE="5" COLOR="GREEN">:-</FONT></TD>
                <TD><INPUT TYPE="password" NAME="pass"></TD>
        </TR>
        </TABLE><BR>
        <INPUT TYPE="submit" name="logbut" value="Sign In">
        </CENTER>
  </FORM>
 </BODY>
"""

class CheckAdminLogin(Resource):
        def __init__(self):
                Resource.__init__(self)

        def getChild (self, path, request):
                user_id=request.args["uid"][0]
                password=request.args["pass"][0]
                self.uid=user_id
                if (user_id=="admin" and password=="password"):
                        return AdminPage()

                else:
                        return AdminFail()




class AdminPage(Resource):
        def __init__(self):
                Resource.__init__(self)
        def render_GET(self, request):
                return """ <HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
        .admin {position:absolute; top:125px; left:41%;}
        .reg {position:absolute; top:120px; left:45%;}
  </style>
</HEAD>
```

```
 <BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
			<H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
			</FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
		<img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
  <div class="reg">
	<div>
		<BR><BR><br>

		<A HREF="/group"><font size="5">Register Group</size></A><BR><BR><br>
		<A HREF="/register"><font size="5">Register Slave</size></A><BR><BR>
	</div>

  </div>
  <div class="admin">
	<B><I><H1><CENTER><U><FONT COLOR="#990033" size="6">Welcome
Admin</FONT></U></CENTER></H1></I></B>
  </div>
 </BODY>"""




class AdminFail(Resource):
	def __init__(self):
		Resource.__init__(self)
	def render_GET(self, request):
		return """
				<HEAD>
  <style type="text/css">
	.rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
	.mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
	.home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
			<H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
			</FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
```

```
             <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><CENTER><B><blink>Wrong username or Password<BR>Please enter
correctly...</blink></B></CENTER><BR><BR>
                                   <CENTER><A HREF="/admin"><font size="5">click here to login
again></font></CENTER>
</body>
                                   """


class IndexForm(Resource):
        def __init__(self):
                Resource.__init__(self)

        def render_GET(self, request):

                return """ <HEAD>
  <style type="text/css">
   .leftsect {float:right; padding-top:100px;padding-right:50px;width:200px;}
   .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
   .mainimage {position:absolute;left:10px;top:130px;padding-left:30px}
   .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
   .wrapper{position:absolute;top:0;left:0;right:0;bottom:0;overflow:hidden;}
   .table {position:absolute; top:130px;left:850px;}
   .a {animation-duration: 1500ms; animation-name: blink; animation-iteration-count: infinite;
animation-direction: alternate;}
   @keyframes blink {
          0% {
            opacity: 1;
            color: pink;
          }
          25% {
            color: green;
            opacity: 0;
          }
          50% {
            opacity: 1;
            color: blue;
          }
          75% {
           opacity: 0;
           color: orange;
          }
          100% {
           opacity: 1;
           color: pink;
          }
         }
  </style>
 </HEAD>
 <BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
                        <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                        </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
               <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
   <div class="mainimage">
               <img src="http://pilf.in/wp-content/uploads/2013/08/mit.jpg" width=700 height=400>
</img>
   </div>
  <div class="table">
  <table width="400px" height="200px" border="1">
            <tr>
            <td>
                        <img src="http://t1.gstatic.com/images?q=tbn:ANd9GcTefTy0Le2WEe-
W_MNO_b43xzqaJnU9U9DiWTNXlciZGIa4kAZuwzfBkSWt" alt="description here" width=200
height=200/>
            </td>
            <td>
                        <div class="a">
                        <H1><CENTER><FONT COLOR="#FF0000" size='4'><blink>User'S
Block</blink></FONT></H1>
                        </div>
                        <CENTER><A HREF="/user"><FONT size='4'>Register
Yourself</FONT></A><BR><BR>
                        <A HREF="/login"><FONT size='4'>Log-in</FONT></A><BR><BR>
                        <A HREF="/reservation"><FONT size='4'>Reserve
Resources</FONT></A><BR></CENTER>
            </td>
            </tr>
            <tr>
            <td>
                        <div class="a">
                        <H1><CENTER><FONT COLOR="#FF0000"
size='4'><blink>Administrator's</blink><BR><blink>Block</blink></FONT></H1>
                        </div>
                        <center><A HREF="/admin"><FONT size='4'>Admin Log-
in</FONT></A></CENTER>
            </td>
            <td>
                        <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcQRsTbb-
0zSkS5tzCXivEkd6Zt08pWuOS_V7UUR9RfUtuLZtGYaJKWMmkg" alt="description here"
bgcolor="#323943" width=200 height=200/>
            </td>
            </tr>
  </table>
  </div>
 </BODY>
"""
```

```
##################################################


class RegisterslaveForm(Resource):
        def __init__(self):
                Resource.__init__(self)

        def render_GET(self, request):
                return """<HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
  </div>
  <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
          <div class="home">
                 <A HREF="/index"><font size="4">Home</font></A>
          </div>
        <BR><BR><BR><BR><CENTER><B><blink>Enter the IP address of
slave...</blink></B></CENTER><BR>
        <CENTER><form action="/registers" >
                        <font color="green" size="4">IP address :</font><input type="text"
name="ip" /> <br>
                        <input type="submit">
        </form></center>
</body>
                        """

class Registers(Resource):
        def __init__(self):
                Resource.__init__(self)

        def render_GET(self, request):
                ip=request.args["ip"][0]
                print ip

                url = "http://" + ip + ":57000/describe"
                print url
                try:
                        response_raw=urllib2.urlopen(url)#.read()
                        print response_raw.info()
                        html = response_raw.read()
                        #return html
                        response = json.loads(html)
                except urllib2.URLError:
                        print ("Could not connect to slave")
                        return """<HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
        <BR><BR><BR><BR><br><br><br><CENTER><B><blink><font color="red"
size="7">Slave cannot be registered...</font></blink></B></CENTER>
</body>"""
                except urllib2.HTTPError:
                        print ("HTTP Error")
                dt=int(response["Disk-Total"])
                da=int(response["Disk-Avail"])
                mt=int(response["MemTotal"])
                cc=int(response["cpu cores"])
                mf=int(response["MemFree"])
                mu=mt-mf
                da=dt-da
                register(ip,dt,da,mt,mu,cc)
                reserv=Reservation_Status.query.filter_by().first()
                reserv.total_cores=reserv.total_cores + cc
                session.commit()
                return """<HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
```

```
                        <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                        </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
                <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
            <br><br><br><br><CENTER><B><blink><font color="red" size="7">Slave
registered...</font></blink></B></CENTER>
            <br><center><font color="#760992">IP address of slave - %s<br>Disk-Total - %d
KB<br>Mem-Total - %d KB<br>Cores - %d<br></font></center>
</body>"""%(ip,dt,mt,cc)



class Inter(Resource):
            def __init__(self):
                        Resource.__init__(self)


            def getChild (self, path, request):
                        print " in inter"
                        if path == "":
                                    print " in path"
                                    opt=request.args["opt"][0]
                                    print opt
                                    if opt == "new":
                                                return AddImageResource()
                                    else:
                                                print "in else"
                                                return ResumptionForm(opt)



class ResumptionForm(Resource):
            def __init__(self, vmid):
                        Resource.__init__(self)
                        self.vmid = vmid
                        self.status = ""

            def render_GET(self, request):
                        instance = Instance.query.filter(Instance.id == self.vmid).first()
                        ready_slaves = Slave.query.filter(Slave.name == instance.slave_name).first()
                        url = "http://" + ready_slaves.ip_address + ":57000/resume"
                        vmname={"vmid":self.vmid}
                        print vmname
                        data = urllib.urlencode(vmname)
                        try:
                                    #response_raw=urllib2.urlopen(url,data).read()
```

```
                    #response = json.loads(response_raw)
                    response_raw=urllib2.urlopen(url,data)#.read()
                    print response_raw.info()
                    html = response_raw.read()
                    #return html
                    response = json.loads(html)
                    print response["status"]
                    if response["status"]=="OK":
                        pass
                except urllib2.URLError:
                    #Slave down
                    print ("Could not connect to slave")
                    return json.dumps("Error-1....please contact admin")
                except urllib2.HTTPError:
                    print ("HTTP Error")
                except:
                    print "Cannot be resumed ...VM needs to be transfered."
                    return json.dumps("Error-2....please contact admin")
                instance = Instance.query.filter(Instance.id == self.vmid).first()
                instance.status = "OK"
                session.commit()
                self.status = "OK"
                reply = {}
                reply["id"] = self.vmid
                reply["status"] = self.status
                status = Reservation_Status.query.filter_by().first()
                status.used_cores += 2
                ready_slaves.used_cores += 2
                return json.dumps(reply)



class LoginForm(Resource):
        def __init__(self):
                Resource.__init__(self)
                self.putChild("checkLogin", CheckLoginResource())

        def render(self, request):
                        return """<HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>
<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
```

```
            <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
 <div class="home">
 <A HREF="/index"><font size="4">Home</font></A>
 </div>
<BR><BR><br><br><br>
<form action="login/checkLogin" method="post">
        <H1><CENTER><U><I><B><FONT SIZE="" COLOR="#996633">User
Login</FONT></B></I></U></CENTER></H1>
                        <center><font color="green" size ="5">User ID :</font><input type="text"
name="user_id" /></center> <br>
                        <center><font color="green" size ="5">Password :</font><input
type="password" name="password" /></center> <br>
                        <!-- Do you have reservation ? : <input type="radio" name="reservation"
value="yes"> Yes

                                        <input type="radio" name="reservation" value="no"> No -->
                        <center><input type="submit"></center>
                        </form>
 </BODY>
                        """

class CheckLoginResource(Resource):
        def render(self, request):
                try:
                        user_id=request.args["user_id"][0]
                        password=request.args["password"][0]
                        user_group = User.query.filter(and_(User.user_id ==
user_id,User.user_password == password) ).first()
                        if (user_group is None):
                                print "redirect to login page"
                                return """
                                <HEAD>
 <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
 </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
  </div>
  <div class="mainlogo">
            <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
 <div class="home">
 <A HREF="/index"><font size="4">Home</font></A>
```

```
  </div>
<BR><BR><BR><BR><CENTER><B><blink>Wrong Login id and password<BR>Please enter
correctly...</blink></B></CENTER><BR><BR>
<form action="/login" method="post">
                                        <center><input type="submit" value="click here to login
again"></center>
                                        </form>
</body>
                                        """
                              else:
                                        print "aps1"
                                        update_instance()
                                        h = int(time.strftime('%H'))
                                        m = int(time.strftime('%M'))
                                        s = int(time.strftime('%S'))
                                        now_time = datetime.datetime(1900,1,1,h,m,s)
                                        #now_time = time.strptime(time.strftime('%X'),"%X")
                                        print now_time
                                        now_date = datetime.date.today()
                                        global user
                                        user = user_group.group_group_name

          reservation=Reservation.query.filter(and_(now_time>Reservation.start_time,now_time<Rese
rvation.end_time,now_date==Reservation.request_date,user_group.group_group_name==Reservation
.owner_group_name)).all()
                                        print reservation
                                        global reserv_flag
                                        reserv_flag = 0
                                        global reserv_id
                                        reserv_id = -1

                                        if reservation is not None:
                                                  print "aps2"
                                                  for r in reservation:
                                                            #print "hii333333"
                                                            requested_cores = int(r.requested_cores)
                                                            used_cores = int(r.used_cores)
                                                            #print requested_cores + "---" + used_cores
                                                            if ((requested_cores-used_cores)>=2):
                                                                      print "hi5"
                                                            #Code for launching image


                                                                      reserv_flag = 1
                                                                      reserv_id=r.reservation_id
                                                                      print r.reservation_id
                                                                      print reserv_id

                                                                      #r.used_cores += 2
                                                                      session.commit()
                                                                      print "Rescheduled value print kar rahe h"
                                                                      print r.rescheduled
                                                                      if r.rescheduled ==1:
                                                                                return """<HEAD>
  <style type="text/css">
```

```
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
  </div>
  <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><CENTER><B><blink>Your time has been
Rescheduled...</blink></B></CENTER><BR>
<CENTER><B>Sorry for the inconvenience. Resources are full at this time.</B></CENTER>
<CENTER><B>You can use the resources after 1 hour.</B></CENTER>
</body> """
                                                        status =
Reservation_Status.query.filter_by().first()
                                                        if status.total_cores-status.used_cores <= 4:
                                                                print "hi4= master- main Check
login resources"

                                                                suspend_vm()


                                                                return """<HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
  </div>
  <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><CENTER><B><blink>You have a
Reservation...</blink></B></CENTER><BR><BR>
 <center><form action="/resume" method="post"> <font color="#760992">Click to Proceed
</font><br> <input type="submit" value="Submit"> </form></center>
</body> """
                                                                                else:
                                                                                        return """ <HEAD>
  <style type="text/css">
          .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
          .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
          .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
                    <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                    </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
            <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><CENTER><B><blink>You have a
Reservation...</blink></B></CENTER><BR><BR>
 <center><form action="/resume" method="post"> <font color="#760992">Click to Proceed
</font><br> <input type="submit" value="Submit"> </form></center>
</body> """

                                                                return """
                                                                <HEAD>
  <style type="text/css">
          .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
          .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
          .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
                    <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                    </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
            <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><CENTER><B><blink>No Reservation Launch Normal Image ...</blink></B></CENTER><BR><BR>
                                <center><form action="/launchimage" method="post">
                                <font color="#760992">To Launch an image click on the button below </font><br>

                                <input type="submit" value="Launch The Image">
                                </form></center>
</body>
                                """

                                else:

          reservation=Reservation.query.filter(now_date==Reservation.request_date).all()
                                          if reservation is not None:
                                                  print "have reservation"
                                          return """
                                          <HEAD>
  <style type="text/css">
          .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
          .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
          .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
                    <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT COLOR="#6CF4EF" size='10'>TCOE
                    </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
            <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><CENTER><B><blink>No Reservation Launch Normal Image ...</blink></B></CENTER><BR><BR>
                                <center><form action="/launchimage" method="post">
                                <font color="#760992">To Launch an image click on the button below </font><br>

                                <input type="submit" value="Launch The Image">
                                </form></center>
```

```
</body>
                """
        except:
            return Exception.message


class ResumeForm(Resource):
    def __init__(self):
        Resource.__init__(self)

    def render(self, request):
        instance = Instance.query.filter(and_(Instance.status=="NOTOK" ,
Instance.user==user)).all()
        options = ""
        for i in instance:
                strid = str(i.id)
                options += """<option value="%s">%s</option>""" %(strid, strid)
        options += """<option value="new">New VM</option>"""
        print options
        return """<HEAD>
  <style type="text/css">
    .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
    .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
    .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
        <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
        </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><CENTER><B><blink>Select an instance or launch new
VM...</blink></B></CENTER><BR><BR>
    <center><font color="#760992">Select one of the options</font>
            <form action="/intermed/" >
            <br>
            <font color="#760992">Options:</font>
                                                                <select
name="opt">%s</select>
                                                                <br>
            <input type="submit">
            </form>        </center>
</body>
            """    %(options)
```

```
def suspend_vm():
	print "i m in sus"
	instance=Instance.query.filter(and_(Instance.status=="OK",Instance.vmtype=="BE")).first()
	if instance is not None:
		name={"instance":instance.id}
		print name
		data = urllib.urlencode(name)

		ready_slaves = Slave.query.filter(Slave.name==instance.slave_name).first()
		#print ready_slaves
		url = "http://" + ready_slaves.ip_address + ":57000/suspend"
		try:
			#response_raw=urllib2.urlopen(url,data).read()
			#response = json.loads(response_raw)

			response_raw=urllib2.urlopen(url,data)#.read()
			print response_raw.info()
			html = response_raw.read()
			#return html
			response = json.loads(html)
		except urllib2.URLError:
			#Slave down
			print ("Could not connect to slave")
		except urllib2.HTTPError:
			print ("HTTP Error")
		if response["status"]=="OK":
			instance.status="NOTOK"
			status = Reservation_Status.query.filter_by().first()
			status.used_cores -= 2
			ready_slaves.used_cores -= 2
			update_instance()				#inserted today
		else:
			print "Instance cannot be destroyed"
		session.commit()
		print "hurray suspended!!!!!!!!!!!!!!!!!!!!!!!!"



def update_instance():
	count_cores=0
	total_used_cores=0
	ready_slaves = Slave.query.all()
	for slave in ready_slaves:
		count_cores += slave.total_cores
		url = "http://" + slave.ip_address + ":57000/check"
		instance=Instance.query.all()
		count1 = 0
		count2 = 0
		for be in instance:
			if (be.status=="OK" and be.slave_name==slave.name):
				bname={"bid":be.id}
				data = urllib.urlencode(bname)
				try:
```

```
                    #response_raw=urllib2.urlopen(url,data).read()
                    #response = json.loads(response_raw)

                    response_raw=urllib2.urlopen(url,data)#.read()
                    print response_raw.info()
                    html = response_raw.read()
                    #return html
                    response = json.loads(html)
                    print response["status"]
                    if response["status"]=="NOTOK":
                        be.status="NOTOK"
                        count1 = count1 +1
                        if be.vmtype == "AR":
                            user = be.user
                            reserv = Reservation.query.filter(and_(datetime.date.today()==Reservation.request_date,Reservation.owner_group_name== user)).first()
                            reserv.used_cores -= 2
                    else:
                        print "not changed"
                        count2 += 1
                    session.commit()
                except urllib2.URLError:
                    print ("Could not connect to slave")
                except urllib2.HTTPError:
                    print ("HTTP Error")
        session.commit()
        count1 = count1 * 2
        total_used_cores += count2 * 2
        slave.used_cores = count2 * 2
        session.commit()

        url = "http://" + slave.ip_address + ":57000/describe"
        try:
            #response_raw=urllib2.urlopen(url).read()
            #response = json.loads(response_raw)

            response_raw=urllib2.urlopen(url)#.read()
            print response_raw.info()
            html = response_raw.read()
            #return html
            response = json.loads(html)

        except urllib2.URLError:
            print ("Could not connect to slave")
        except urllib2.HTTPError:
            print ("HTTP Error")
        dt=int(response["Disk-Total"])
        da=int(response["Disk-Avail"])
        mt=int(response["MemTotal"])
        cc=int(response["cpu cores"])
        mf=int(response["MemFree"])
        mu=mt-mf
        da=dt-da
        slave.total_ram=mt
        slave.used_ram=mu
        slave.total_hdd_size=dt
        slave.used_hdd_size=da
        session.commit()
    status = Reservation_Status.query.filter_by().first()
    status.total_cores=count_cores
    status.used_cores = total_used_cores
    session.commit()


class LaunchImageForm(Resource):
    def __init__(self):
        Resource.__init__(self)
        self.putChild("addImage", AddImageResource())
    def render(self, request):
        #try:
        for time in range(0,5):
            print "time is %s"%time
            if(launchtime(time)):
                hours = time
            else:
                    hours = time - 1
                    break
        if(hours == -1):
            return "not enough capacity"
        else:
            a = int(hours)+1
            #print a #change 1
            opts_hr = ""
            #print hours #change 2
            for i in range(1,a):
                opts_hr += """<option value="%d">%d</option>""" %(i, i)
            return"""
            <HEAD>
  <style type="text/css">
    .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
    .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
    .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
        <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
        </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
      <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><CENTER><B><blink>How much time you want to use
VM...</blink></B></CENTER><BR><BR>
        <center><font color="#760992">Select the number of hours you want to run your
image:</font>
                        <form action="/resume" method="post">
                        <br>
                        <font color="#760992">Number of hours:</font>
                                                                                        <select
name="hours">%s</select>
                                                                                        <br>
                        <input type="submit">
                        </form></center>
</body>
                        """     %(opts_hr)


def launchtime(time1):
                        h = int(time.strftime('%H')) + int( time1 )
                        #print "hours inside loop %s"%h
                        m = int(time.strftime('%M'))
                        s = int(time.strftime('%S'))
                        now_time = datetime.datetime(1900,1,1,h,m,s)
                        now_date = datetime.date.today()
                        reservation =
Reservation.query.filter(and_(now_time>Reservation.start_time,now_time<Reservation.end_time,no
w_date==Reservation.request_date)).all()
                        x = 0
                        if reservation is not None:
                                        for r in reservation:
                                                x= x + (r.requested_cores - r.used_cores)
                        #print x
                        status = Reservation_Status.query.filter_by().first()
                        total_cores = int(status.total_cores)
                        used_cores = int(status.used_cores)
                        available_cores = (total_cores - (used_cores + x))
                        if ( available_cores >= 2 ):
                                print " IF  - return true %s"%available_cores
                                return True
                        else:
                                print "ELSE - return false %s"%available_cores
                                return False

class AddImageResource(Resource):
        def render(self, request):
                image_id = str(DiskImage.query.first().id)
                return """
                                                <HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>
```

```
<BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
  </div>
  <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
 <div class="home">
 <A HREF="/index"><font size="4">Home</font></A>
 </div>
<br><br><br><br><br>
<CENTER><B><blink>Image ready...</blink></B></CENTER><BR><BR>
<center><a href="/image/%s/launch"><font color="#760992">Click here to launch image ubuntu-
jdk</font></a></center>
</body>        """%(image_id)

class ReservationForm(Resource):
        def __init__(self):
                Resource.__init__(self)
                self.putChild("addReservation", AddReservationResource())

        def render(self, request):
                        opts_hr = ""
                        for i in range(1,5):
                                opts_hr += """<option value="%d">%d</option>""" %(i, i)
                        opts_time = ""
                        for i in range(9,18):
                                opts_time += """<option value="%d">%d</option>""" %(i, i)
                        return """
                        <HEAD>
                         <style type="text/css">
                          .leftsect {position:absolute; left:700px; padding-top:70px;padding-
right:50px;width:200px;}
                          .rightsect {position:absolute;left:10px;top:10px;padding-
left:150px;width:1195px}
                          .mainimage {position:absolute;left:10px;top:130px;padding-left:30px}
                          .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
                          .wrapper{position:absolute;top:0;left:0;right:0;bottom:0;overflow:hidden;}
                         </style>
                         </HEAD>

                 <BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
  </div>
  <div class="mainlogo">
```

```
            <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
  <div class="mainimage">
            <img src="http://pilf.in/wp-content/uploads/2013/08/mit.jpg" width=700 height=400>
</img>
  </div>
<div class="leftsect">
<form action="reservation/addReservation" method="post">
        <pre>
        <FONT size='4'>YOU CAN RESERVE ONLY FOR TOMORROW...</FONT>
        <FONT size='4'>MAXIMUM RESERVATION TIME IS 4 HRS</FONT>
        <FONT size='4'>U CAN RESERVE FROM 9 AM TO 5 PM</FONT>
        <FONT size='4'>OWNER :</FONT>                                   <input
type="text" name="reservation_owner" />
        <FONT size='4'>PASSWORD :</FONT>                                    <input
type="password" name="owner_password" />
        <FONT size='4'>PLATFORM VALUE : </FONT>                          <input
type="text" name="pfv" />
        <FONT size='4'>DO YOU WANT TO REGISTER AS SLAVE (0:NO , 1: YES) :
</FONT><input type="text" name="ras" />
        <FONT size='4'>RAM :</FONT>
<input type="text" name="r" />
        <FONT size='4'>MEMBER OR NOT ( 0:NO , 1 YES) : </FONT>
<input type="text" name="mon" />
        <FONT size='4'>NUMBER OF CORES :</FONT>
<input type="text" name="number_cores" />
        <FONT size='4'>NUMBER OF HOURS:</FONT><select name="hours">%s</select>
        <FONT size='4'>RESERVATION TIME:(24 hr format) :</FONT><select
name="start_time">%s</select>
                <input type="submit">
        </pre>
</form>
</DIV>
</BODY>
                        """ %(opts_hr,opts_time)
#                       Compromise : <input type="radio" name="compromise" value="yes"> Yes
<input type="radio" name="compromise" value="no"> No

class AddReservationResource(Resource):
        def render(self, request):

                user_group = Group.query.filter(and_(Group.group_name ==
request.args["reservation_owner"][0],Group.group_password ==
request.args["owner_password"][0])).first()
                a = datetime.datetime(1900,1,1,int(request.args["start_time"][0]))
                b = datetime.datetime(1900,1,1,int(request.args["start_time"][0]) +
int(request.args["hours"][0]))
                if (user_group is not None):

        reservation=Reservation.query.filter(and_(or_(and_(a>Reservation.start_time,a<Reservation.
end_time) ,
and_(b>Reservation.start_time,b<Reservation.end_time)),datetime.date.today()+datetime.timedelta(1)
==Reservation.request_date))
                        print reservation
            if reservation is not None:
                x=0
                print "Hiiiiiiiiiiiiiiiiii"
                max_cores=Reservation_Status.query.first()
                for r in reservation:
                    x+=r.requested_cores
                print x
                temp1 = int(max_cores.total_cores)
                temp2 = int(request.args["number_cores"][0])

                if ((temp1-x)>temp2):
                    temp =
Reservation.query.order_by(desc(Reservation.reservation_id)).first()
                    print temp

                    if temp is None:
                        ID = 1
                    else:
                        ID = temp.reservation_id + 1

        #           print ID
                    ram_v = int(request.args["r"][0])
                    exec_v= int(request.args["hours"][0])*60
                    use_cores = int(request.args["number_cores"][0])

    owner=Group.query.filter_by(group_name=request.args["reservation_owner"][0]).first()

                    prior_val =
round((int(request.args["pfv"][0])+int(request.args["ras"][0])-math.log(ram_v)/math.log(2)-
math.log(exec_v)+int(request.args["mon"][0])),2)
                    prior_value= 10 + 0.25*prior_val
                    start_time =
datetime.datetime(1900,1,1,int(request.args["start_time"][0]))
                    end_time =
datetime.datetime(1900,1,1,int(request.args["start_time"][0]) + int(request.args["hours"][0]))
                    time_int=int(request.args["hours"][0])
                    max_cores=Reservation_Status.query.first()
                    max_core = int(max_cores.total_cores)

    reservation_schedule=ReservationSchedule.query.filter(and_(start_time==ReservationSchedu
le.start_time)).first()
                    print "About reservation schedule-------------------------------"
                    print reservation_schedule   #for loop lagana hoga for r in
reservation:
                    if reservation_schedule is not None:
                        #print reservation_schedule.id
                        print "there are reservations at this time"

    reserv_schedule=ReservationSchedule.query.filter(and_(start_time==ReservationSchedule.sta
rt_time)).all()
                        print "#########All Res_sched################"
                        for rs in reserv_schedule:
                            pntr = rs.id - 1
                            print rs.id
                        prior2_chk =
Reservation.query.filter(and_(start_time==Reservation.start_time)).order_by((Reservation.priority_va
lue)).all()
                        for prior_chk in prior2_chk:
                            if prior_chk.rescheduled==0:
                                break
                        print prior_chk.priority_value
                        print "#########All Res_sched################"
                        global core_minus
                        use_core=use_cores+ rs.used_cores
                        if(use_core > max_core):
                            if(prior_chk.priority_value>=prior_value):
                                return """ <HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
            <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
            </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
         <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><br><CENTER><B><blink>Sorry no more resources available at this
time...</blink></B></CENTER><BR>
<CENTER><B>Please click below link to reserve again at different time</B>
<br><br><A HREF="/reservation"><FONT size='4'>Reserve
Resources</FONT></A><BR></CENTER>

</body>
                            """
                            else:
                                global pval
                                global resid
                                global reqcores
                                global usecores
                                global statime
                                global endtime
                                global reqdate
                                global owner1
                                pval = prior_chk.priority_value
                                reqcores =
prior_chk.requested_cores
```

```
                                                                usecores = prior_chk.used_cores
                                                                statime = prior_chk.start_time
                                                                endtime = prior_chk.end_time
                                                                reqdate = prior_chk.request_date
                                                                owner1 = prior_chk.owner
                                                                prior_chk.start_time
=prior_chk.start_time+datetime.timedelta(hours=1)
                                                                prior_chk.end_time
=prior_chk.end_time+datetime.timedelta(hours=1)
                                                                prior_chk.rescheduled=2
                                                                prior_chk1 = Reservation.query.all()
                                                                for rs1 in prior_chk1:
                                                                        print rs1.reservation_id
                                                                resid =rs1.reservation_id +2
                                                                reservation =
Reservation(pval,resid,reqcores,usecores,statime,endtime,reqdate,owner1,1,"none")
                                                                session.commit()
                                                                change_resv=
ReservationSchedule.query.filter(and_(ReservationSchedule.id==prior_chk.reservation_id)).first()
                                                                print change_resv.id

        change_resv.start_time=change_resv.start_time+datetime.timedelta(hours=1)

        change_resv.end_time=change_resv.end_time+datetime.timedelta(hours=1)
                                                                use_core=use_core-
change_resv.cores_required
                                                                #rs.used_cores=rs.used_cores-
core_minus
                                                print use_cores
                                                print use_core
                                                print rs.used_cores
                                                reservation_schedule =
ReservationSchedule(start_time,end_time,time_int,max_core,use_core,use_cores)
                                                session.commit()
                                        else:
                                                print "there are no reservations at this time"
                                                use_core=use_cores
                                                print use_core
                                                reservation_schedule =
ReservationSchedule(start_time,end_time,time_int,max_core,use_core,use_core)
                                                session.commit()
                                        reservation =
Reservation((prior_value),ID,int(request.args["number_cores"][0]),0,start_time,end_time,datetime.dat
e.today()+datetime.timedelta(1),owner,0,"none")
                                        session.commit()
                                        return """<HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
```

```
                        <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                        </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
               <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><br><CENTER><B><blink>Reservation
confirmed...</blink></B></CENTER><BR><BR>

</body>"""
                                                else:
                                                            print "false"
                                                            return """<HEAD>
  <style type="text/css">
            .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
            .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
            .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
                        <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                        </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
               <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><br><CENTER><B><blink>Reservation cannot be
satisfied...</blink></B></CENTER><BR><BR>

</body>"""
                        else:
                                    return """<HEAD>
  <style type="text/css">
            .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
            .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
            .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>
```

```
<BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
  </div>
  <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
 <div class="home">
 <A HREF="/index"><font size="4">Home</font></A>
 </div>
<BR><BR><BR><BR><CENTER><B><blink>Wrong Login id or password<BR>Please enter
correctly...</blink></B></CENTER><BR><BR>
</body> """


class UserForm(Resource):
        def __init__(self):
                Resource.__init__(self)
                self.putChild("addUser", AddUserResource())

        def render(self, request):
                        group=Group.query.all()
                        options = ""
                        for i in group:
                                strid = str(i.group_name)
                                options += """<option value="%s">%s</option>""" %(strid, strid)
                        print options

                        return """ <form action="user/addUser" method="post">
                        Id :<input type="text" name="user_id" /> <br>
                        Password :<input type="password" name="user_password" /> <br>
                        Group:  <select name="opt">%s</select> <br>
                        <input type="submit">
                        </form> """ %(options)

class AddUserResource(Resource):
        def render(self, request):
                grp=str(request.args["opt"][0])
                group=Group.query.filter_by(group_name=grp).first()
                print group
                usr=str(request.args["user_id"][0])
                passwd=str(request.args["user_password"][0])
                user = User(usr,passwd,group)
                session.commit()
                return """<HEAD>
 <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
 </style>
```

```
</HEAD>

<BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
  </div>
  <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
 <div class="home">
 <A HREF="/index"><font size="4">Home</font></A>
 </div>
<BR><BR><BR><BR><br><CENTER><B><blink>User
added...</blink></B></CENTER><BR><BR>

</body>"""

class GroupForm(Resource):
        def __init__(self):
                Resource.__init__(self)
                self.putChild("addGroup", AddGroupResource())

        def render(self, request):
                        return """
                        <HEAD>
 <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
        .useradd {position:absolute;top:40%; left:43%;}
 </style>
</HEAD>
<BODY bgcolor="#C6C1C9">
  <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
  </div>
  <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
  </div>
 <div class="home">
 <A HREF="/index"><font size="4">Home</font></A>
 </div>
<BR><BR><br><br><br>
<H1><CENTER><U><I><B><FONT SIZE="" COLOR="#996633">Add
Group</FONT></B></I></U></CENTER></H1>
```

```
<form action="group/addGroup" method="post">
        <div class="useradd">
                        <font color="green" size ="5">Group Name :</font><input type="text"
name="group_name" /> <br>
                        <font color="green" size ="5">Password :</font><input type="password"
name="group_password" /> <br>
                        <font color="green" size ="5">Group Owner:</font>    <input type="text"
name="group_owner" /><br>
                        <input type="submit">
        </div>
</form>
 </BODY>

                        """
class AddGroupResource(Resource):
        def render(self, request):
#               group=Group.query.filter_by(group_name=request.args["group"][0]).first()
                group =
Group(request.args["group_name"][0],request.args["group_owner"][0],request.args["group_password
"][0])
                session.commit()
                return """<HEAD>
  <style type="text/css">
        .rightsect {position:absolute;left:10px;top:10px;padding-left:150px;width:1195px}
        .mainlogo {position:absolute;left:10px;top:10px;padding-left:25px}
        .home {position:absolute; top:120px; left:35px;}
  </style>
</HEAD>

<BODY bgcolor="#C6C1C9">
   <div class="rightsect" >
                <H1><CENTER><marquee bgcolor="#474644" size='60'><FONT
COLOR="#6CF4EF" size='10'>M<FONT COLOR="#FC26AA" size='10'>i<FONT
COLOR="#6CF4EF" size='10'>TCOE
                </FONT><FONT COLOR="#FC26AA" size='6'> VCL<FONT><FONT
COLOR="#02C81D" size='8'><sup>+</sup></FONT></marquee></CENTER></H1>
   </div>
   <div class="mainlogo">
          <img src="http://t3.gstatic.com/images?q=tbn:ANd9GcSCDHz3jw2-
n13gZYtz2C2nRzH0m9tWlp9WJFBTcD3gUvxg0zqb4SilAg" width=100 height=100> </img>
   </div>
  <div class="home">
  <A HREF="/index"><font size="4">Home</font></A>
  </div>
<BR><BR><BR><BR><br><CENTER><B><blink>Group
added...</blink></B></CENTER><BR><BR>

</body> """

class SlavesResource(Resource):
        def __init__(self):
                Resource.__init__(self)

        def render_GET(self, request):
                session.commit()
        slaves = Slave.query.all()
        count = 1
        reply = {"slaves" : {}}
        for slave in slaves:
            slave_name = "slave-" + str(count)
            reply["slaves"][slave_name] = {}
            reply["slaves"][slave_name]["name"] = slave.name
            reply["slaves"][slave_name]["status"] = slave.status
            count += 1
        return json.dumps(reply)

class SlaveRootResource(Resource):
    def __init__(self):
        Resource.__init__(self)

    def render(self, request):
        return "Wrong slave name"

    def getChild(self, path, request):
        slave = Slave.query.filter_by(name=path).first()
        if slave is None:
            return self
        else:
            return SlaveResource(slave)

class SlaveResource(Resource):
    def __init__(self, slave):
        Resource.__init__(self)
        self.slave = slave

    def render_GET(self, request):
        reply = {}
        reply["name"] = self.slave.name
        reply["status"] = self.slave.status
        return json.dumps(reply)

class DiskImageRootResource(Resource):

    def __init__(self, dhcp_server, host_factory):
        Resource.__init__(self)
        self.dhcp_server = dhcp_server
        self.host_factory = host_factory

    def getChild(self, path, request):
        image = DiskImage.query.filter_by(id=path).first()
        if image is None:
            return self
        else:
            return DiskImageResource(image, self.dhcp_server, self.host_factory)


class DiskImageResource(Resource):
    def __init__(self, image, dhcp_server, host_factory):
        Resource.__init__(self)
        self.image = image
        self.host_factory = host_factory
        self.dhcp_server = dhcp_server

    def render_GET(self, request):
        reply = {}
        reply["id"] = self.image.id
        reply["name"] = self.image.name
        reply["description"] = self.image.description
        return json.dumps(reply)

    def getChild (self, path, request):
        if path == "launch":
            cores = self.image.default_cores
            hdd_size = self.image.default_hdd_size
            ram = self.image.default_ram
            #Add code to handle missing default_cores in self.image
            params = request.args
            if "cores" in params:
                cores = params["cores"][0]
            if "hdd_size" in params:
                hdd_size = params["hdd_size"][0]
            if "ram" in params:
                ram = params["ram"][0]
            session.commit()
            print "in launch"
            ready_slaves = Slave.query.order_by(Slave.used_cores).all()
    #master statement
            #print ready_slaves
            instance = None
            for slave in ready_slaves:
                if slave.has_resources (cores=cores,ram=ram, hdd_size=hdd_size):
                    print ("Selected slave %s" %slave.ip_address)
                    url = "http://" + slave.ip_address + ":57000/launchimage"
                    print url
                    host = self.host_factory.generate_host()

                    while True:
                        #id = "i-%x" % random.randint(0x1000000,
0xfffffff)
                        id = "i-ubuntu-%x" % random.randint(0x100, 0xfff)
                        print "instance id"
                        print id
                        instance = Instance.query.filter_by(id=id).first()
                        if instance is None:
                            break
                    if reserv_flag == 0:
                        vmtype="BE"
                    else:
                        print reserv_id

    reservation=Reservation.query.filter(Reservation.reservation_id==reserv_id).first()
                        vmtype="AR"
                        reservation.used_cores += 2
                    instance = Instance(id=id , cores=cores, hdd_size=hdd_size,
ram=ram,

    kernel=self.image.default_kernel, ramdisk=self.image.default_ramdisk,

    disk=self.image, host=host, slave=slave, vmtype=vmtype, user=user)
                    os.system("ip addr add dev eth0 %s/32"
%(instance.host.ip_address))
                    os.system("iptables -t nat -A PREROUTING -d %s/32 -j
DNAT --to-destination %s"%(instance.host.ip_address, instance.host.private_ip))
                    os.system("iptables -t nat -A OUTPUT -d %s/32 -j DNAT --
to-destination %s" %(instance.host.ip_address, instance.host.private_ip))
                    slave.decrease_resources(cores=cores, ram=ram,
hdd_size=hdd_size)
                    self.dhcp_server.add_host(host)
                    session.commit()
                    print("-------> kernel id %s" %self.image.default_kernel.id)
                    print("-------> instance id %s" %instance.id)
                    params = {"disk_image_id":self.image.id, "cores":cores,
"ram":ram, "hdd_size":hdd_size,

    "kernel_id":self.image.default_kernel.id, "ramdisk_id":self.image.default_ramdisk.id,

    "mac_address":host.mac_address, "ip_address":host.ip_address, "instance_id":instance.id }


                    status = Reservation_Status.query.filter_by().first()
                    status.used_cores += 2
                    session.commit()

                    data = urllib.urlencode(params)
                    print data
                    try:
                        print "hello"
                        try:
                            #response_raw = urllib2.urlopen(url,
data).read()
                            response_raw = urllib2.urlopen(url,
data)#.read()
                            print response_raw.info()
                            html = response_raw.read()
                            #return html
                            #response = json.loads(html)
                            response = json.loads(html)
                        except ValueError:
                            print ("Invalid JSON returned by slave.
Trying new slave")
                            continue
                        if response["status"] == "OK":
                            print "Status OK"
                            return InstanceResource(instance)

                    except urllib2.URLError:
                        #Slave down
                        print ("Could not connect to slave")
                        session.rollback()
                        continue
                    except urllib2.HTTPError:
                        print ("HTTP Error")
                        session.rollback()

                        continue

            return InstanceResource(None)
        elif path == "download":
            print (self.image.id)
            return static.File(os.path.abspath(".") + "/images/disks/" + self.image.id)



class InstanceRootResource(Resource):
    def __init__(self):
        Resource.__init__(self)

    def getChild(self, path, request):
        instance = Instance.query.filter_by(id=path).first()
        if instance is None:
            return self
        else:
            return InstanceResource(instance)


class InstanceResource(Resource):
    def __init__(self, instance):
        Resource.__init__(self)
        #self.instance = Resource
        self.instance = instance

    def render_GET(self, request):
        reply = {}
        if self.instance is not None:
            try:    #
                reply["id"] = self.instance.id
                reply["public_ip_address"] = self.instance.host.ip_address
                reply["private_ip_address"] = self.instance.host.private_ip
                reply["mac_address"] = self.instance.host.mac_address
                return json.dumps(reply)
            except: #err
                print "Instance Running"  #
                return json.dumps("Instance Running")#
        else:
            print "Resources are full..."
            print self.instance
            return json.dumps("Error-3....contact admin!!!!")


if __name__ == '__main__':
    config = ConfigObj("../../cloud43.conf")
    print "Hello"
    #Start DHCP server
```

```
            base_path = os.curdir
            #public_ips = config["MASTER"]["DHCP"]["PUBLICIPS"]
            #ip_range_min = public_ips[0:public_ips.find("-")]
            #ip_range_max = public_ips[public_ips.find("-")+1:]
            config["MASTER"]["DHCP"]["PUBLIC_IP_MIN"] = "10.1.44.220"
            config["MASTER"]["DHCP"]["PUBLIC_IP_MAX"] = "10.1.44.235"
            config["MASTER"]["DHCP"]["PRIVATE_IP_MIN"] = "10.1.44.236"
            config["MASTER"]["DHCP"]["PRIVATE_IP_MAX"] = "10.1.44.250"
            config.write()
            dhcp_server = DHCPServer(
                                                        daemon =
config["MASTER"]["DHCP"]["DAEMON"],
                                                        path= base_path + "/dhcp",
                                                        interface = "eth0",
#config["MASTER"]["DHCP"]["INTERFACE"],
                                                        subnet =
config["MASTER"]["DHCP"]["SUBNET"],
                                                        netmask =
config["MASTER"]["DHCP"]["NETMASK"],
                                                        ip_range_min =
config["MASTER"]["DHCP"]["PRIVATE_IP_MIN"],
                                                        ip_range_max =
config["MASTER"]["DHCP"]["PRIVATE_IP_MAX"],
                                                        routers = "192.168.1.10",
                                                        subnet_mask = "255.255.255.0",
                                                        broadcast_address = "192.168.1.255",
                                                        dns =
config["MASTER"]["DHCP"]["DNS"])
            dhcp_server.start()
            host_factory = HostFactory(config)
            config["dhcp_server"] = dhcp_server
            config["host_factory"] = host_factory
            config.write()
            #Start twisted.web.reactor to start receiving HTTP requests
            root = RootResource(config)
            site = Site(root)
            reactor.listenTCP(8000, site)
            reactor.run()
            os.system("iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE")

            #os.system("iptables -t nat -A PREROUTING -d 179.254.169.254/32 -p tcp -m tcp -j DNAT -
-to-destination 172.16.41.200")
            dhcp_server.kill_daemon()
            sleep(1)
```

### File “models.py”

```
'''
This module contains all the models
'''
from elixir import *
from threading import Thread
import simplejson as json
```

```
#import zmq
import threading
from sqlalchemy import and_
import os
import sys
from time import sleep
import signal
import random
import math
from configobj import ConfigObj
#from sqlalchemy.test.schema import table_name

DEFAULT_HDD_SIZE = 5*1024*1024                    #size in KBS
DEFAULT_CORES = 1                                                                           #size in
KBS
DEFAULT_RAM = 512*1024

metadata.bind = 'sqlite:///db/cloud43.db'

def make_empty_file(filename):
    fd = open(filename, "w")
    fd.truncate(0)
    fd.close()
class Reservation_Status(Entity):

    using_options(tablename="reservation_status")

    total_cores=Field(Integer)
    used_cores=Field(Integer)

    def __init__(self, total_cores, used_cores):
        self.total_cores=total_cores
        self.used_cores=used_cores

    def update_used_cores(self,used_cores):
        self.used_cores+=used_cores
        session.commit()
class ReservationSchedule(Entity):
    using_options(tablename="reservation_schedule")

    start_time=Field(DateTime)
    end_time=Field(DateTime)
    time_interval=Field(Integer)
    total_cores=Field(Integer)
    used_cores=Field(Integer)
    cores_required=Field(Integer)

    def __init__(self,start_time,end_time,time_interval,total_cores,used_cores,cores_required):
        self.start_time=start_time
        self.end_time=end_time
        self.time_interval=time_interval
        self.total_cores=total_cores
        self.used_cores=used_cores
        self.cores_required=cores_required
```

```
class Reservation(Entity):
    using_options(tablename="reservation")

    reservation_id=Field(Integer,primary_key=True)
    requested_cores=Field(Integer)
    used_cores=Field(Integer)
    owner=ManyToOne("Group")
    start_time=Field(DateTime)
    end_time=Field(DateTime)
    request_date = Field(Date)
    priority_value=Field(Integer)
    rescheduled =Field(Integer)
    description=Field(String(30))

    def
__init__(self,priority_value,reservation_id,requested_cores,used_cores,start_time,end_time,request_d
ate,owner,rescheduled,description=None):
        self.priority_value=priority_value
        self.reservation_id=reservation_id
        self.requested_cores=requested_cores
        self.used_cores=used_cores
        self.start_time=start_time
        self.end_time=end_time
        self.request_date=request_date
        self.owner=owner
        self.rescheduled=rescheduled
        self.description=           description

    def update_used_cores(self,used_cores):
        self.used_cores+=used_cores
        session.commit()

class User(Entity):

    using_options(tablename="users")

    user_id = Field(String(30), primary_key=True)
    user_password = Field(String(30))
    group = ManyToOne("Group")

    def __init__(self,user_id,user_password,group):
        self.user_id=user_id
        self.user_password=user_password
        self.group=group

class Group(Entity):

    using_options(tablename="groups")

    group_name=Field(String(30),primary_key=True)
    group_owner=Field(String(30))
    group_password=Field(String(30))

    def __init__(self,group_name,group_owner,group_password):
        self.group_name=group_name
```

```
		self.group_owner=group_owner
		self.group_password=group_password

class Kernel(Entity):
	'''
	Represents an kernel
	'''
	using_options(tablename="kernels")

	id = Field(String(30), primary_key=True)
	name = Field(String(30))
	description = Field(String(30))

	def __init__(self, id, name, description):
		self.id = id
		self.name = name
		self.description = description

class RAMDisk(Entity):
	'''
	Represents an RAM Disk
	'''
	using_options(tablename="ramdisks")

	id = Field(String(30), primary_key=True)
	name = Field(String(30))
	description = Field(String(30))

	def __init__(self, id, name, description):
		self.id = id
		self.name = name
		self.description = description

class DiskImage(Entity):
	'''
	Represents an image
	'''
	using_options(tablename="diskimages")

	id = Field(String(30), primary_key=True)
	name = Field(String(30))
	description = Field(String(30))
	default_kernel = ManyToOne("Kernel")
	default_ramdisk = ManyToOne("RAMDisk")
	default_ram = Field(Integer)				#size in KBs
	default_cores = Field(Integer)
	default_hdd_size = Field(Integer)			#size in KBs

	def __init__(self, id, name, description, default_kernel=None,
							default_ramdisk=None,
default_hdd_size=DEFAULT_HDD_SIZE,
							default_cores=DEFAULT_CORES,
default_ram=DEFAULT_RAM):
		self.id = id
		self.name = name
		self.description = description
		self.default_kernel = default_kernel
		self.default_ramdisk = default_ramdisk
		self.default_hdd_size = default_hdd_size
		self.default_cores = default_cores
		self.default_ram = default_ram

class Slave(Entity):

	using_options(tablename="slaves")

	name = Field(String(30),primary_key=True)
	password = Field(String(50))
	ip_address = Field(String(15))
	total_ram = Field(Integer)				#size in KBs
	total_cores = Field(Integer)
	total_hdd_size = Field(Integer)
	used_ram = Field(Integer)				#size in KBs
	used_cores = Field(Integer)
	used_hdd_size = Field(Integer)
	status = Field(String(10))

	def __init__(self, name, password, ip_address, total_ram, total_cores, total_hdd_size,
							used_ram=0, used_cores=0,
used_hdd_size=0):
		self.name = name
		self.password = password
		self.ip_address = ip_address
		self.total_ram = total_ram
		self.total_cores = total_cores
		self.total_hdd_size = total_hdd_size
		self.used_ram = used_ram
		self.used_cores = used_cores
		self.used_hdd_size = used_hdd_size

	def __repr__(self):
		return "<Slave %s>" %(str(self.__dict__))

	def __eq__(self, other) :
		return self.name == other.name

	def has_resources (self, cores, hdd_size, ram):
		print "in has resources"
		print "self.used_cores"
		print self.used_cores
		print cores
		print self.total_cores
		print self.used_ram
		print ram
		print self.total_ram
		print self.used_hdd_size
		print hdd_size
		print self.total_hdd_size
		#return (self.used_cores + cores <= self.total_cores) and (self.used_ram + ram <=
self.total_ram) and (self.used_hdd_size + hdd_size <= self.total_hdd_size)
```

```
		return (self.used_cores + cores <= self.total_cores) and (self.used_ram + 0 <= 
self.total_ram) and (self.used_hdd_size + 0 <= self.total_hdd_size)

	def decrease_resources (self, cores, hdd_size, ram):
		self.used_cores += cores
		self.used_hdd_size += hdd_size
		self.used_ram += ram

class Instance(Entity):
	using_options(tablename="instances")

	id = Field(String(30), primary_key=True)
	disk = ManyToOne("DiskImage")
	kernel = ManyToOne("Kernel")
	ramdisk = ManyToOne("RAMDisk")
	slave = ManyToOne("Slave")
	ram = Field(Integer)				#size in KBs
	cores = Field(Integer)
	hdd_size = Field(Integer)
	has_one('host', of_kind='Host', inverse='instance')
	status = Field(String(10))
	vmtype = Field(String(10))
	user = Field(String(30))

	def __init__(self, id, disk, kernel, ramdisk, slave, ram, cores, hdd_size, host, vmtype, user):
		self.id = id
		self.disk = disk
		self.kernel=kernel
		self.ramdisk=ramdisk
		self.slave=slave
		self.ram = ram
		self.cores = cores
		self.hdd_size = hdd_size
		self.host = host
		self.vmtype = vmtype
		self.status = "OK"
		self.user = user

class PersistentInstance(Entity):
	using_options(tablename="persistentinstances")

	id = Field(String(30), primary_key=True)
	disk = ManyToOne("PersistentDiskImage")
	kernel = ManyToOne("Kernel")
	ramdisk = ManyToOne("RAMDisk")
	slave = ManyToOne("Slave")
	ram = Field(Integer)				#size in KBs
	cores = Field(Integer)
	hdd_size = Field(Integer)
	has_one('host', of_kind='Host', inverse='instance')
	data = Field(String(100))

	def __init__(self, id, disk, kernel, ramdisk, slave, ram, cores, hdd_size, host, data=None):
		self.id = id
		self.disk = disk
```

```
        self.kernel=kernel
        self.ramdisk=ramdisk
        self.slave=slave
        self.ram = ram
        self.cores = cores
        self.hdd_size = hdd_size
        self.host = host
        self.data = data


class Host(Entity):
    using_options(tablename="hosts")

    mac_address = Field(String)
    private_ip = Field(String)
    public_ip = Field(String)
    belongs_to('instance', of_kind='Instance')

    def __init__(self, mac_address, private_ip, ip_address):
        self.mac_address = mac_address
        self.private_ip = private_ip
        self.ip_address = ip_address

class PersistentImageMacBinding(Entity):
    using_options(tablename="persistentbind")
    belongs_to('persistent_disk_image', of_kind='PersistentDiskImage')
    mac_address = Field(String(20))
    instance_id = Field(String(20))

    def __init__(self, persistent_disk_image, mac_address, instance_id):
        self.persistent_disk_image = persistent_disk_image
        self.mac_address = mac_address
        self.instance_id = instance_id

class PersistentDiskImage(Entity):
    using_options(tablename="persistentimages")
    id = Field(String(30), primary_key=True)
    name = Field(String(30))
    description = Field(String(30))
    default_kernel = ManyToOne("Kernel")
    default_ramdisk = ManyToOne("RAMDisk")
    default_ram = Field(Integer)                                #size in KBs
    default_cores = Field(Integer)
    default_hdd_size = Field(Integer)                   #size in KBs
    has_one('mac_binding', of_kind='PersistentImageMacBinding',
inverse='persistent_disk_image')

    def __init__(self, id, name, description, default_kernel=None,
                            default_ramdisk=None,
default_hdd_size=DEFAULT_HDD_SIZE,
                            default_cores=DEFAULT_CORES,
default_ram=DEFAULT_RAM):
        self.id = id
        self.name = name
        self.description = description
        self.default_kernel = default_kernel
        self.default_ramdisk = default_ramdisk
        self.default_hdd_size = default_hdd_size
        self.default_cores = default_cores
        self.default_ram = default_ram


class HostFactory():
    def __init__(self, config):
        self.config = config

    def _random_mac(self):
        mac = [ 0x00, 0x16, 0x3e,
        random.randint(0x00, 0x7f),
        random.randint(0x00, 0xff),
        random.randint(0x00, 0xff) ]
        return ':'.join(map(lambda x: "%02x" % x, mac))

    def _number_to_ip(self, num):
        d = num % 256
        c = int(num / 256) % 256
        b = int(num / (256 ** 2)) % 256
        a = int(num / (256 ** 3)) % 256
        return ".".join(map(str, [a, b, c, d]))

    def _ip_to_number(self, ip):
        octets = map(int,ip.split("."))
        num = (256 ** 3) * octets[0] + (256 ** 2) * octets[1] + 256 * octets[2] + octets[3]
        return num

    def generate_persistent_mac(self):
        while True:
            mac_address = self._random_mac()
            duplicate = Host.query.filter_by(mac_address=mac_address).first()
            if duplicate is None:
                duplicate = 
PersistentImageMacBinding.query.filter_by(mac_address=mac_address).first()
                if duplicate is None:
                    break
        return mac_address

    def generate_host(self):
        mac_address = self.generate_persistent_mac()
        public_ip_min = 
self._ip_to_number(self.config["MASTER"]["DHCP"]["PUBLIC_IP_MIN"])
        public_ip_max = 
self._ip_to_number(self.config["MASTER"]["DHCP"]["PUBLIC_IP_MAX"])

        private_ip_min = 
self._ip_to_number(self.config["MASTER"]["DHCP"]["PRIVATE_IP_MIN"])
        private_ip_max = 
self._ip_to_number(self.config["MASTER"]["DHCP"]["PRIVATE_IP_MAX"])

        hosts = Host.query.all()
        host = None
        done = False
        for public_num in range(public_ip_min, public_ip_max + 1):
            for private_num in range(private_ip_min, private_ip_max + 1):
                ip_address = self._number_to_ip(public_num)
                private_ip = self._number_to_ip(private_num)
                dups = filter(lambda host: host.ip_address == ip_address or
host.private_ip == private_ip, hosts)
                if len(dups) == 0:
                    host = Host(mac_address=mac_address,
ip_address=ip_address, private_ip=private_ip)
                    done = True
                    break
            if done:
                break
        return host

    def generate_ip_pair(self):
        pass
#   def __cmp__ (self, other):
#       if(self.public_ip == other.public_ip and self.mac_address == other.mac_address and
self.private_ip == other.private_ip):
#           return 0
#       else:
#           return -1

class DHCPServer:
    def __init__(self, daemon, path, interface, subnet, netmask, ip_range_min, ip_range_max,
routers, subnet_mask, broadcast_address, dns):
        #Load default values for missing ones
        if(daemon is None or len(daemon) == 0):
            daemon = "/usr/bin/dhcpd"
        if(path is None):
            path = ""
        if(interface is None or len(interface) == 0):
            interface = "eth0"
        if(subnet is None or len(subnet) == 0):
            subnet = "192.168.0.0"

        self.daemon = daemon
        if not os.path.exists(path):
            try:
                os.mkdir(path)
                os.chmod(path, 0777)
            except:
                print "Could not create directory %s. Exiting" %(path)
                sys.exit(1)

        self.config = path + "/cloud43-dhcp.conf"
        self.leases = path + "/cloud43-dhcp.leases"
        self.trace = path + "/cloud43-dhcp.trace"
        self.pid = path + "/cloud43-dhcp.pid"

        make_empty_file(self.config)
        make_empty_file(self.leases)
        os.chmod(self.leases, 0666)
```

```
		self.daemon = daemon
		self.interface = interface
		self.subnet = subnet
		self.netmask = netmask
		self.ip_range_min = ip_range_min
		self.ip_range_max = ip_range_max
		self.routers = routers
		self.subnet_mask = subnet_mask
		self.broadcast_address = broadcast_address
		self.dns = dns
		self.hosts = []
		self.daemon_process = None

	def start(self):
		#Placebo function
		pass

	def _write_config(self):
		with open(self.config, "w") as cfile:
			cfile.write("ddns-update-style none;\n")
			cfile.write("log-facility local7;\n")
			cfile.write("authoritative;\n")
			cfile.write("default-lease-time 86400;\nmax-lease-time 86400;\n")

			cfile.write("subnet %s netmask %s {\n" %(self.subnet , self.netmask))

			cfile.write("\trange %s %s;\n" %(self.ip_range_min , self.ip_range_max))
			cfile.write("\toption routers %s;\n" %(self.routers))
			cfile.write("\toption subnet-mask %s;\n" %(self.subnet_mask))
			cfile.write("\toption broadcast-address %s;\n" %(self.broadcast_address))
			cfile.write("\toption domain-name-servers %s;\n" %(self.dns))
			cfile.write("}\n")

			for host in self.hosts:
				cfile.write("host %s {\n" %("node-" + host.private_ip))
				cfile.write("\thardware ethernet %s;\n" %(host.mac_address))
				cfile.write("\tfixed-address %s;\n" %(host.private_ip))
				cfile.write("}\n")

	def add_host(self, host):
		#Check duplicate MAC entry. Ignoring if there are any
		#duplicate = filter(lambda address: address == host, self.hosts)
		#session.commit()
		#duplicate = Host.query.filter_by(mac_address=host.mac_address).first()
		#if duplicate is not None:
		#	print "Duplicate host. Ignoring it"
		#	return
		self.hosts.append(host)
		self._write_config()
		self.restart_daemon()


	def remove_host(self, host):
		addr = filter(lambda address: address == host, self.hosts)
```

```
        if(len(addr) != 0):
            self.hosts.remove(addr[0])
            self._write_config()
            self.restart_daemon()
        else:
            #IP not found. Ignoring
            #TODO:Add logging behaviour
            pass

    def restart_daemon(self):
        self.kill_daemon()
        #First argument is sent to the program as the name with which it was invoked with
        args = [self.daemon, "-cf", self.config, "-lf", self.leases, "-pf", self.pid, "-tf",
self.trace, self.interface]
        os.spawnvp (os.P_NOWAIT, self.daemon, args)
        #Sleep to allow dhcp to start
        sleep (2)
        print("Daemon restarted")

    def kill_daemon(self):
        if os.path.exists(self.pid):
            try:
                os.kill(int(open(self.pid).read()), signal.SIGKILL)
                #Sleep to allow dhcp to stop
                sleep (1)
            except OSError as (strerror):
                print "Could not kill previous dhcp daemon:", strerror
            except:
                print "Unexpected error:", sys.exc_info()[0]
setup_all(create_tables=True,)

#kernel = Kernel ("eki-7654321", "eki-7654321", "")
#ramdisk = RAMDisk ("eri-7654321", "er-7654321", "")
#disk = PersistentDiskImage (id="emi-7654321", name="emi-7654321", description="",
default_kernel=kernel,
#
    default_ramdisk = ramdisk, default_ram=524288, default_hdd_size=5242880,
default_cores=1)
#config = ConfigObj("../../cloud43.conf")
#bind = PersistentImageMacBinding (persistent_disk_image = disk,
mac_address=HostFactory(config).generate_persistent_mac())
#session.commit()
```

## File “RegisterSlave.py”

```
import hashlib
import sqlite3
import os
import sys
import random
import string
```

```
conn=sqlite3.connect('db/cloud43.db')
c=conn.cursor()
DATABASE_OPTIONS = {'timeout': 30}

def register(ip,dt,da,mt,mu,cc):

    random_password=random.random()
    random_password=str(random_password)

    pwd_md5=hashlib.md5()
    pwd_md5.update(random_password)
    password=pwd_md5.hexdigest()
    name="slave_"
    flag=True
    #print random_password


    while flag:
        slave_postfix=random.randint(0, 99)
        temp_name='slave_'+str(slave_postfix)
        print temp_name
        c.execute("select * from slaves where name='%s'" %temp_name)
 #c.execute("select * from Slaves where SlaveUserName='%s'" %temp_name)
 #c.fetchall()
         #print c.rowcount
        if not c.fetchone():
            name=temp_name
          flag=False

    #print temp_name
    #print password

    c.execute("insert into slaves values('%s','%s','%s','%d','%d','%d','%d','%d','%d','%s') "
%(name,password,ip,mt,cc,dt,mu,0,da,""))
    conn.commit()
    print("Successfully inserted")
    c.close()
```

## 8.2 Slave’s Code

### File “ main.py”

```
Main script which runs on the Slave
"""

import sys
import simplejson as json
from models import ImageStore, Instance, generate_libvirt_xml
import os
path = '.'
if path not in sys.path:
    sys.path.append(path)
import libvirt
```

```
from twisted.web.resource import Resource
from twisted.web.server import Site
from twisted.internet import reactor, defer

total_cores = 4
total_ram = 4 * 1024 * 1024 * 1024
total_hdd_size = 50 * 1024 * 1024
used_cores = 0
used_ram = 0
used_hdd_size = 0

class RootResource(Resource):
        def __init__(self):
                Resource.__init__(self)
                self.putChild("launchimage", LaunchImageResource())
                self.putChild("launchpimage", LaunchPersistentImageResource())
                self.putChild("stoppimage", StopPersistentImageResource())
                self.putChild("check", CheckAvail())
                self.putChild("suspend", Suspend_vm())
                self.putChild("resume", ResumeVm())
          self.putChild("describe",Describeslave())

class Describeslave(Resource):
        def __init__(self):
                Resource.__init__(self)
                self.defered = defer.Deferred()

        def render(self, request):
                fileptr=os.popen("df | egrep \"/dev/sda1\"")
                var=fileptr.readline()
                tokens=var.split()
                disktotal=tokens[1]#4#tokens[1]
                diskavail=tokens[3]#3#tokens[3]
          fileptr=os.popen( "cat /proc/meminfo | egrep \"MemTotal\"")
          var=fileptr.readline()
          tokens=var.split()
          memtotal=tokens[1]
                fileptr=os.popen( "cat /proc/meminfo | egrep \"MemFree\"")
          var=fileptr.readline()
          tokens=var.split()
          memfree=tokens[1]
          fileptr=os.popen( "cat /proc/cpuinfo | egrep \"cpu cores\"")
          var=fileptr.readline()
          tokens=var.split()
          core=tokens[3]
                var=fileptr.readline()
          tokens=var.split()
          core1=tokens[3]
          core2=int(core) + int(core1)
                return json.dumps({"Disk-Total" :disktotal,"Disk-
Avail":diskavail,"MemTotal":memtotal,"MemFree":memfree,"cpu cores":core2})



class ResumeVm(Resource):
	def __init__(self):
		Resource.__init__(self)
		self.defered = defer.Deferred()

	def render(self, request):
		print "in resumption process now..."
		vmname = request.args
		vmid = str(vmname["vmid"][0])
		print vmid

		instance_path = "%s/instances/%s" %(os.path.abspath("."),vmid)
		libvirt_xml_file = open(instance_path + "/libvirt.xml", "r")
		libvirt_xml = libvirt_xml_file.read()

		libvirt_conn = libvirt.open("qemu:///system")
		libvirt_conn.createXML(libvirt_xml, 0)
		dom = "OK"
		return json.dumps({"status" : dom})


class CheckAvail(Resource):
	def __init__(self):
		Resource.__init__(self)
		self.defered = defer.Deferred()

	def render(self, request):
		x="check-hi"
		print x
		bname = request.args
		bnm = str(bname["bid"][0])
		print bnm
		libvirt_conn = libvirt.open("qemu:///system")
		try:
			dom1=libvirt_conn.lookupByName(bnm)
			dom="OK"
		except:
			dom="NOTOK"
		print dom
		return json.dumps({"status" : dom})

class Suspend_vm(Resource):
	def __init__(self):
		Resource.__init__(self)
		self.defered = defer.Deferred()

	def render(self, request):
		x="check-vm"
		print x
		vm_name = request.args

		vm = str(vm_name["instance"][0])
		print vm
		libvirt_conn = libvirt.open("qemu:///system")
		try:
			dom1=libvirt_conn.lookupByName(vm)
```

```
            dom1.destroy()
            dom="OK"
        except:
            dom="NOTOK"
        print dom
        return json.dumps({"status" : dom})


class LaunchImageResource(Resource):
    def __init__(self):
        Resource.__init__(self)
        self.defered = defer.Deferred()

    def render(self, request):
        #Check if slave has enough resources
        global total_cores, total_hdd_size, total_ram
        global used_cores, used_hdd_size, used_ram
        global image_store
        params = request.args
        requested_cores = int(params["cores"][0])
        requested_ram = int(params["ram"][0])
        requested_hdd_size = int(params["hdd_size"][0])
        requested_disk_image_id = params["disk_image_id"][0]
        requested_ramdisk_id = params["ramdisk_id"][0]
        requested_kernel_id = params["kernel_id"][0]
        instance_id = params["instance_id"][0]
        mac_address = params["mac_address"][0]
        ip_address = params["ip_address"][0]
        #print (requested_kernel_id + " " + requested_ramdisk_id + " " +
requested_disk_image_id)
        if (requested_cores + used_cores <= total_cores) and (requested_hdd_size +
used_hdd_size <= total_hdd_size) and (requested_ram + used_ram <= total_ram):
            instance_path = "%s/instances/%s" %(os.path.abspath("."), instance_id)
            os.mkdir(instance_path)
            print ("kernel")
            kernel = image_store.clone_kernel(requested_kernel_id, instance_path +
"/kernel")
            print ("ramdisk")
            ramdisk = image_store.clone_ramdisk(requested_ramdisk_id, instance_path
+ "/ramdisk")
            print ("disk")
            disk_image = image_store.clone_disk_image(requested_disk_image_id,
instance_path + "/disk")
            #COnvert 12 and 7620 to request ram and hdd size
            command = "perl " + os.path.abspath("./tools/partition2disk") + " " +
instance_path + "/disk" + " 512 7620"
            os.system(command)
            instance = Instance (id=instance_id, kernel=kernel, ramdisk=ramdisk,
disk_image=disk_image,

    mac_address=mac_address, ip_address=ip_address, log_path="%s/console.log"
%(instance_path), ram=requested_ram,

    cores=requested_cores)
            libvirt_xml = generate_libvirt_xml (instance)
            libvirt_xml_file = open(instance_path + "/libvirt.xml", "w")
            libvirt_xml_file.write(libvirt_xml)
            libvirt_xml_file.close()
            libvirt_conn = libvirt.open("qemu:///system")
            libvirt_conn.createXML(libvirt_xml, 0)
            node = libvirt_conn.listDomainsID()
            print node
            status = "OK"
            status_description = "Instance started"

        else:
            status = "NOTOK"
            status_description = "Not enough resources"
        return json.dumps({"status" : status, "desc" : status_description})

class LaunchPersistentImageResource(Resource):
    def __init__(self):
        Resource.__init__(self)
        self.defered = defer.Deferred()

    def render(self, request):
        #Check if slave has enough resources
        print (request.args)
        global total_cores, total_hdd_size, total_ram
        global used_cores, used_hdd_size, used_ram
        global image_store
        params = request.args
        requested_cores = int(params["cores"][0])
        requested_ram = int(params["ram"][0])
        requested_hdd_size = int(params["hdd_size"][0])
        requested_disk_image_id = params["disk_image_id"][0]
        requested_ramdisk_id = params["ramdisk_id"][0]
        requested_kernel_id = params["kernel_id"][0]
        instance_id = params["instance_id"][0]
        mac_address = params["mac_address"][0]
        #print (requested_kernel_id + " " + requested_ramdisk_id + " " +
requested_disk_image_id)
        if (requested_cores + used_cores <= total_cores) and (requested_hdd_size +
used_hdd_size <= total_hdd_size) and (requested_ram + used_ram <= total_ram):
            instance_path = "%s/pinstances/%s" %(os.path.abspath("."), instance_id)
            if not os.path.exists(instance_path):
                os.mkdir(instance_path)

            print ("kernel")
            instance_kernel_path = instance_path + "/kernel"
            if not os.path.exists(instance_kernel_path):
                kernel = image_store.clone_kernel(requested_kernel_id,
instance_kernel_path)
            else:
                kernel = image_store.get_persistent_kernel(requested_kernel_id,
instance_kernel_path)

            print ("ramdisk")
            instance_ramdisk_path = instance_path + "/ramdisk"
            if not os.path.exists(instance_ramdisk_path):
                ramdisk = image_store.clone_ramdisk(requested_ramdisk_id, 
instance_ramdisk_path)
            else:
                ramdisk = 
image_store.get_persistent_ramdisk(requested_ramdisk_id, instance_ramdisk_path)

            print ("disk")
            instance_disk_image_path = instance_path + "/disk"
            if not os.path.exists(instance_disk_image_path):
                disk_image = 
image_store.clone_disk_image(requested_disk_image_id, instance_disk_image_path)
                #Convert 12 and 7620 to request ram and hdd size
                command = "perl " + os.path.abspath("./tools/partition2disk") + " " + 
instance_disk_image_path + " 512 7620"
                os.system(command)
            else:
                disk_image = 
image_store.get_persistent_disk_image(requested_disk_image_id, instance_disk_image_path)

            instance = Instance (id=instance_id, kernel=kernel, ramdisk=ramdisk, 
disk_image=disk_image, 

    mac_address=mac_address, log_path="%s/console.log" %(instance_path), 
ram=requested_ram, 

    cores=requested_cores)
            libvirt_xml = generate_libvirt_xml (instance)
            libvirt_xml_file = open(instance_path + "/libvirt.xml", "w")
            libvirt_xml_file.write(libvirt_xml)
            libvirt_xml_file.close()

            libvirt_conn = libvirt.open("qemu:///system")
            libvirt_conn.createXML(libvirt_xml, 0)
            status = "OK"
            status_description = "Persistent Instance started"

        else:
            status = "NOTOK"
            status_description = "Not enough resources"
        return json.dumps({"status" : status, "desc" : status_description})

class StopPersistentImageResource(Resource):
    def __init__(self):
        Resource.__init__(self)
        self.defered = defer.Deferred()

    def render(self, request):
        print ("destroy called")
        instance_id = request.args["instance_id"][0]
        libvirt_conn = libvirt.open("qemu:///system")
        libvirt_node = libvirt_conn.lookupByName(instance_id)
        libvirt_node.destroy()
        return json.dumps ({"status" : "OK" , "desc" : "Instance destroyed"})

if __name__ == '__main__':
    pwd = os.path.abspath(".")
    image_store = ImageStore(pwd + "/store")
    root = RootResource()
    site = Site(root)
    reactor.listenTCP(57000, site)
    print("hello")
    reactor.run()
    print("hello")
```

### File “models.py”

```
import os
import urllib2
master_url = "http://192.168.1.10:8000"

def download_cloud43_image(url, file_path):#, user_name, password):
        #Setup basic authentication handlers
        #http_password_mgr = urllib2.HTTPPasswordMgrWithDefaultRealm()
        #http_password_mgr.add_password(None, url, user_name, password)
        #auth_handler = urllib2.HTTPBasicAuthHandler(http_password_mgr)
        #opener = urllib2.build_opener(auth_handler)
        #urllib2.install_opener(opener)

        #Start request
        print ("downloading " + url + " to " + file_path)
        request = urllib2.urlopen(url)
        print ("opened")
        #Setup file i/o objects
        chunk_size = 4*1024             #4KB
        with open(file_path, "w") as file:
                while True:
                        print ("start chunk")
                        chunk = request.read (chunk_size)
                        print ("end chunk")
                        if not chunk:
                                break
                        file.write(chunk)
        print ("done download")

def generate_libvirt_xml(instance):
        xml = """<domain type='kvm'>
        <name>%s</name>
        <os>
                <type>hvm</type>
                <kernel>%s</kernel>
                <initrd>%s</initrd>
                <cmdline>root=/dev/sda1 console=ttyS0</cmdline>
        </os>
        <features>
                <acpi/>
        </features>
        <memory>%s</memory>
        <vcpu>%s</vcpu>
```

```
    <devices>
        <emulator>/usr/bin/kvm</emulator>
        <disk type='file'>
            <source file='%s'/>
            <target dev='sda'/>
        </disk>
        <interface type='bridge'>
            <source bridge='virbr0'/>
            <mac address='%s'/>
            <model type='e1000'/>
        </interface>



        <network>
                <ip address='%s' netmask='255.255.255.0'>
                </ip>
        </network>
        <graphics type='vnc' port='-1' autoport='yes'>
        </graphics>


        <serial type='file'>
            <source path='%s'/>
            <target port='1'/>
        </serial>
    </devices>
</domain>""" %(instance.id, instance.kernel.path, instance.ramdisk.path, instance.ram, instance.cores, instance.disk_image.path, instance.mac_address, instance.ip_address, instance.log_path)
    return xml


class DiskImage:
    def __init__(self, id, path):
        self.id = id
        self.path = path

class Kernel:
    def __init__(self, id, path):
        self.id = id
        self.path = path

class RAMDisk:
    def __init__(self, id, path):
        self.id = id
        self.path = path

class Instance:
    def __init__(self, id, kernel, ramdisk, disk_image, mac_address, ip_address, log_path, ram, cores):
        self.id = id
        self.kernel = kernel
        self.ramdisk = ramdisk
        self.disk_image = disk_image
        self.mac_address = mac_address
        self.ip_address = ip_address
        self.log_path = log_path
        self.ram = ram
        self.cores = cores

class ImageStore:
    def __init__(self, store_path):
        if store_path[len(store_path) - 1] == "/":
            store_path = store_path[0:len(store_path) - 1]
        self.store_path = store_path

    def _get_disk_image(self, disk_id):
        if not os.path.exists(self.store_path + "/disk/" + disk_id):
            save_path = self.store_path + "/disk/" + disk_id
            print ("down i")
            download_cloud43_image(master_url + "/image/%s/download" %disk_id,
save_path)
        image = DiskImage(disk_id, self.store_path + "/disk/" + disk_id)
        return image

    def _get_kernel(self, kernel_id):
        if not os.path.exists(self.store_path + "/kernel/" + kernel_id):
            save_path = self.store_path + "/kernel/" + kernel_id
            print ("down k")
       print (master_url + "/kernel/%s/download" %kernel_id)
       print (""+ save_path)
            download_cloud43_image(master_url + "/kernel/%s/download" %kernel_id,
save_path)
        kernel = Kernel(kernel_id, self.store_path + "/kernel/" + kernel_id)
        return kernel

    def _get_ramdisk(self, ramdisk_id):
        if not os.path.exists(self.store_path + "/ram/" + ramdisk_id):
            save_path = self.store_path + "/ram/" + ramdisk_id
            print ("down r")
            download_cloud43_image(master_url + "/ramdisk/%s/download"
%ramdisk_id, save_path)
        ramdisk = RAMDisk(ramdisk_id, self.store_path + "/ram/" + ramdisk_id)
        return ramdisk

    def clone_disk_image(self, disk_image_id, save_path):
        disk_image = self._get_disk_image(disk_image_id)
        command = "cp -a %s %s" %(disk_image.path, save_path)
        os.system(command)
        disk_image.path = save_path
        return disk_image

    def clone_kernel(self, kernel_id, save_path):
        kernel = self._get_kernel(kernel_id)
        command = "cp -a %s %s" %(kernel.path, save_path)
        os.system(command)
        kernel.path = save_path
        return kernel
```

```
def clone_ramdisk(self, ramdisk_id, save_path):
        ramdisk = self._get_ramdisk(ramdisk_id)
        command = "cp -a %s %s" %(ramdisk.path, save_path)
        os.system(command)
        ramdisk.path = save_path
        return ramdisk

def get_persistent_disk_image(self, disk_image_id, save_path):
        disk_image = self._get_disk_image(disk_image_id)
        disk_image.path = save_path
        return disk_image

def get_persistent_kernel(self, kernel_id, save_path):
        kernel = self._get_kernel(kernel_id)
        kernel.path = save_path
        return kernel

def get_persistent_ramdisk(self, ramdisk_id, save_path):
        ramdisk = self._get_ramdisk(ramdisk_id)
        ramdisk.path = save_path
        return ramdisk
```

# Chapter IX

# VALIDATION OF DESIGN/SOFTWARE/PROCESS (TESTING)

## 9.1 Introduction

Verification and validation testing are two important tests which are carried out on a software before it has been handed over to the customer. The aim of both verification and validation is to ensure that the product is made according to the requirements of the client, and does indeed fulfill the intended purpose. While verification is a quality control process, the quality assurance process carried out before the software is ready for release is known as validation testing. Its goal is to validate and be confident about the product or system, and that it fulfills the requirements given by the customer. The acceptance of the software from the end customer is also its part. Often, testing activities are introduced early in the software life cycle.

Validation testing provides answers to questions such as:

- Does the software fulfill its intended use?
- Is the company building the right product?
- Can the project be properly implemented?
- Are the documents in line with the development process?

Similarly, various other testing techniques needs to be performed to verify the validation and verification of the software that is to be built. Testing needs to be performed from the starting itself , ,during the requirement gathering. Today, there are various testing tools available in the market using which various spefications of the software can be tested.Thus, our project we have perform some of the testing techniques to verfiy the quality of the software being produced.

## 9.2 Test Specifications

Testing provides a road map that describes the steps to be conducted as a part of testing, when theses steps are planned and then undertaken, and how much effort, time , and resources will be required. This section probides an overview of the entire test document.This document describes both the test plan and the test procedure.

## 9.2.1 Test Plan

### 9.2.1.1 Goals and Objectives

Following are some of the important goals and objectives of the entire test process:

- To find out if each one of the requirements that have been mentioned in the document are met completely.
- To find out if all the funcationalities mentioned in the SRS have been performed.
- To check whether all possible cases for various different modules have been handled and taken care of.

- To find out errorsdue to certain specific positions and inputs and rectify them.

To ensure that the application avoids unacceptable behaviour under normal conditions and behaves properly across a variety of conditions and inputs,normal and abnormal.

To ensure that the program recovers gracefully and quickly from anomalous behviour or conditions.

### 9.2.1.2 Scope

The scope of the entire test process includes extensive unit testing, integration testing, alpha testing and validation testing of the various different modules.It also involves the preparation of various different modules .It also involves the preparation of various different test reports and documents.

### 9.2.1.3 Suspension criteria for testing

Testing be stopped when all the possible cases for a particular functionality will be handled and no defects are reported.

### 9.2.2 Test Deliverables

Test Plan

Test case specifications

Test Design specifications

Unit Test Document

## 9.3 Test Tasks

**Validation Testing**

It is the process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements or not.

The various components tested are as follows :

- For login facility, user name should contain alphabets only.
- In the reservation form wherever required the date and time should be in proper format.
- The reservation date should not be a past date.
- Resources have to be in numbers, alphabets are not allowed.

Validation activities can be divided into the following :

- Low level testing
    - ✓ Unit testing
    - ✓ Integration testing
- High level testing
    - ✓ System testing

### 9.3.1 Unit Testing

In computer programming, unit testing is a procedure used to validate that individual units of source code are working properly. A unit is the smallest testable part of an application.In procedural programming a unit may be  an individual program, function, procedure,etc. While in object-oriented programming, the smallest unit is a method, which may belong to a basic/super class, abstract class or derived/child class.

## 9.4 Test Cases

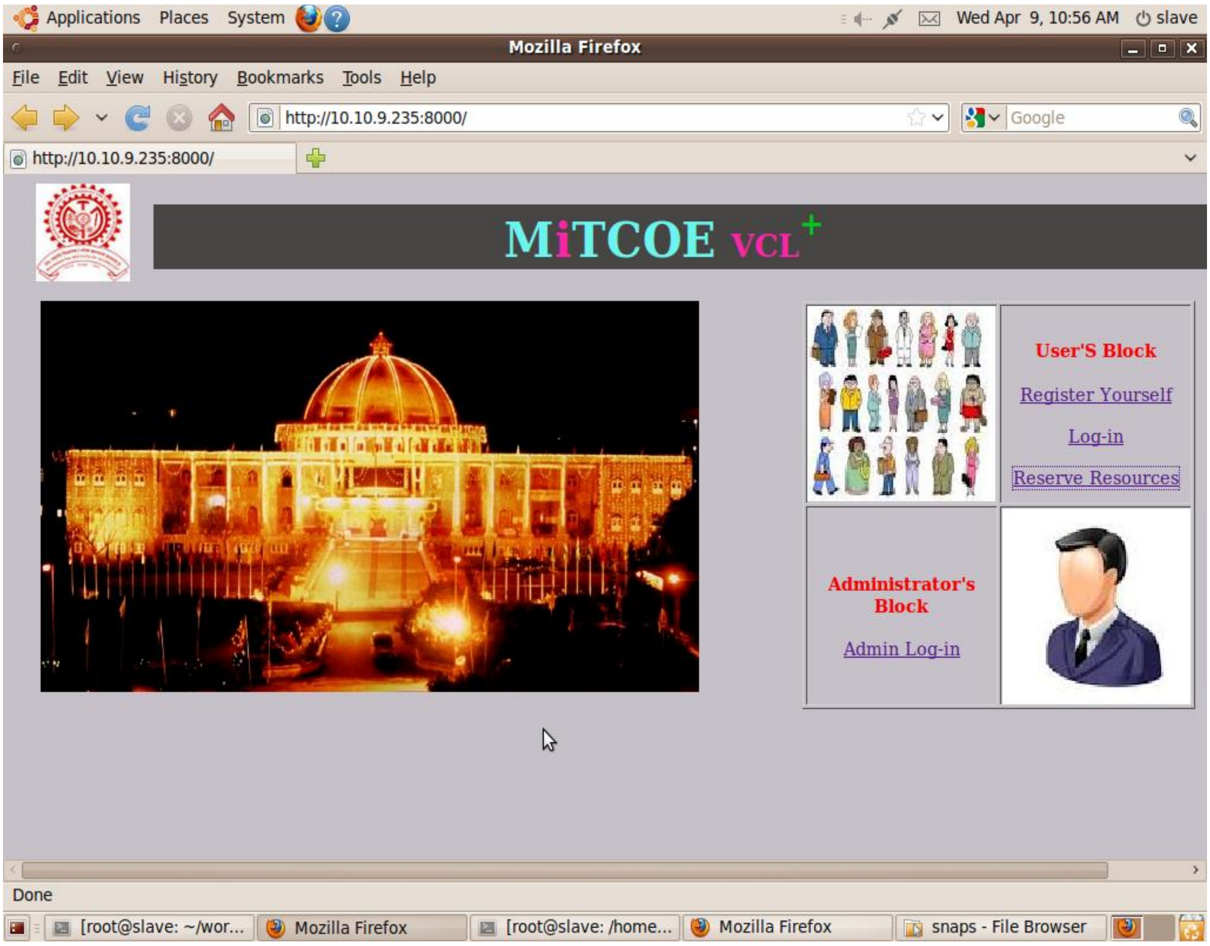
| TEST CASES | | | |
|---|---|---|---|
| **Sr.No** | **Test Conditions** | **Expected Results** | **Actual Result** |
| 1 | To provide correct login facility | User should be logged in after entering correct username and password | Same as expected. |
| 2 | To register the slave | The slave should be permanently registered in the database after entering its correct ip address | Same as expected. |
| 3 | To launch the image | User request for image should be mapped appropriate slave provided enough resources are available. | Same as expected. |
| 4 | To select proper number of hours for running the image | User should be allowed to select hours between 1 to 5 | Same as expected. |
| 5 | To check the sufficiency of resources. | User should be notified about insufficiency of resources when enough resources are not available for launching the image. | Same as expected. |
| 6 | To check if the image is normal image or has reservation. | The requested image should be distinguished as normal or reserved on the basis of reservation time. | Same as expected. |
| 7 | To launch the image with reservation at the request time. | The running virtual machine should be suspended if enough resources are not available for request with reservation. | Same as expected. |

# Chapter X

# RESULT/ANALYSIS/DISCUSSION

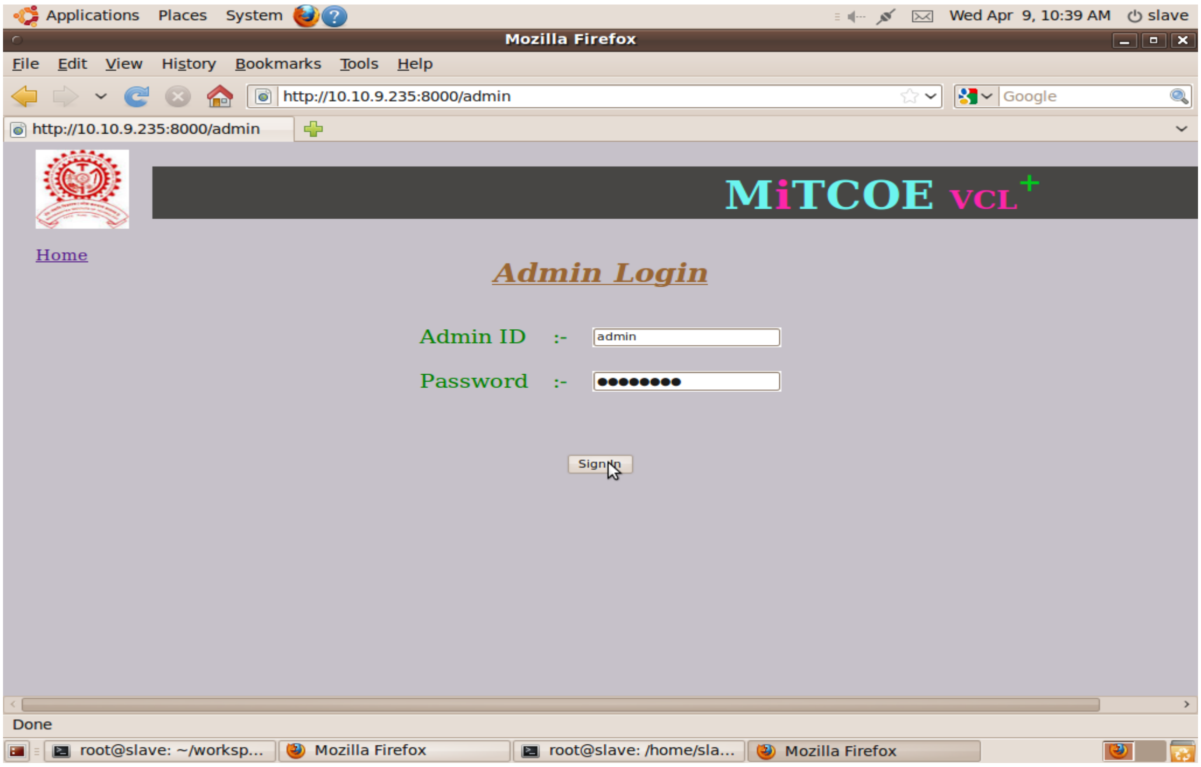
### The “Home Screen” for the VCL look likes as follows:

The URL contains the IP address of the master machine.



To register Slave it needs following steps:

- Admin Login ( username : admin ,password : password )
- Click on “ Register Slave”
- Enter the IP address of the machine which want to register itself as a slave
- Click on “Submit “
- It will get registered as a slave.

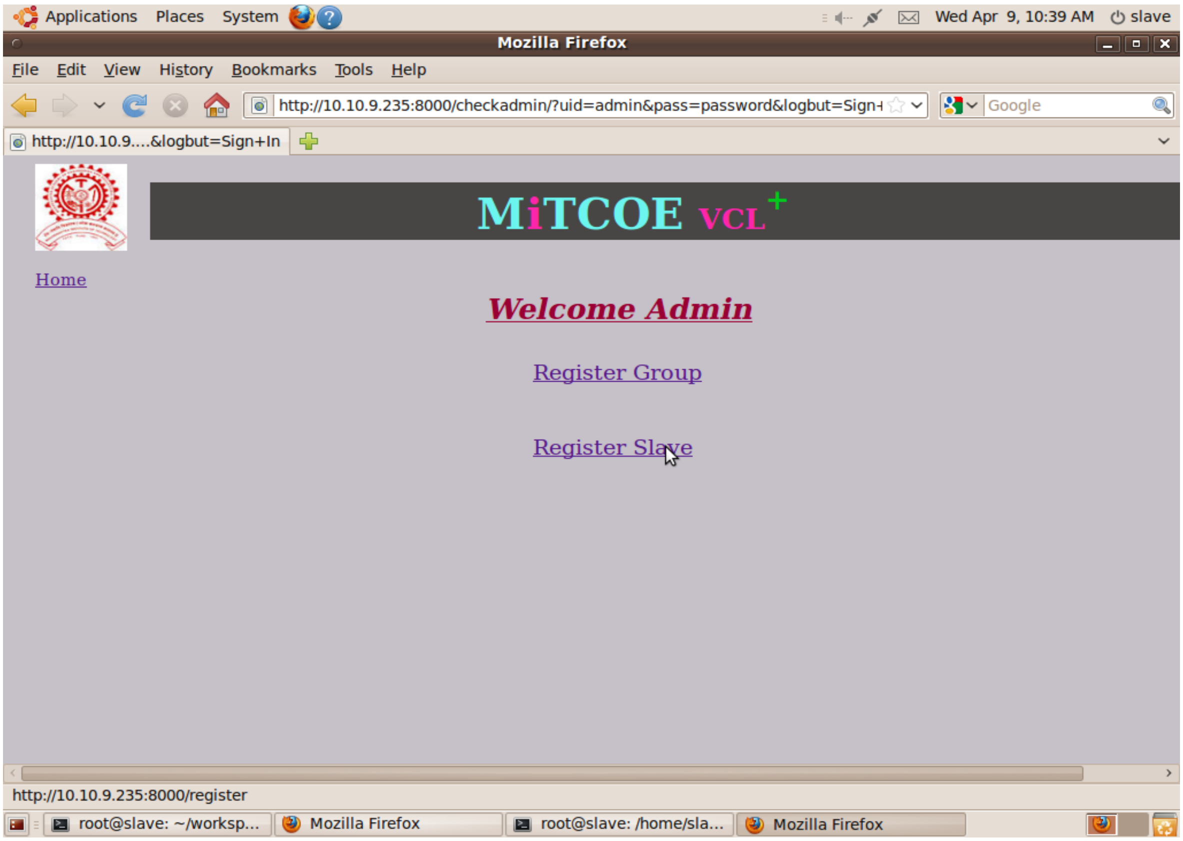


http://10.10.9.235:8000/admin

Admin Login

Admin ID :- admin

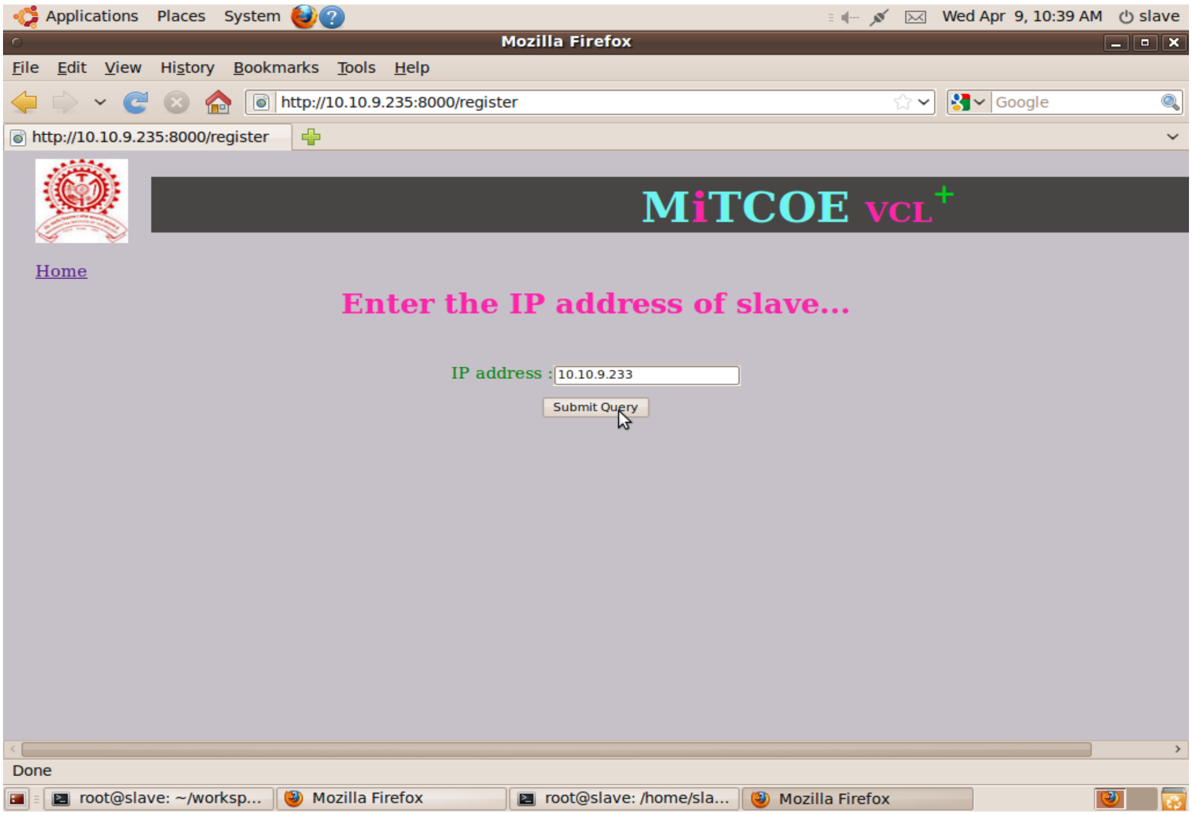
Password :- ●●●●●●●●●



http://10.10.9.235:8000/checkadmin/?uid=admin&pass=password&logbut=Sign+

Welcome Admin

http://10.10.9.235:8000/register

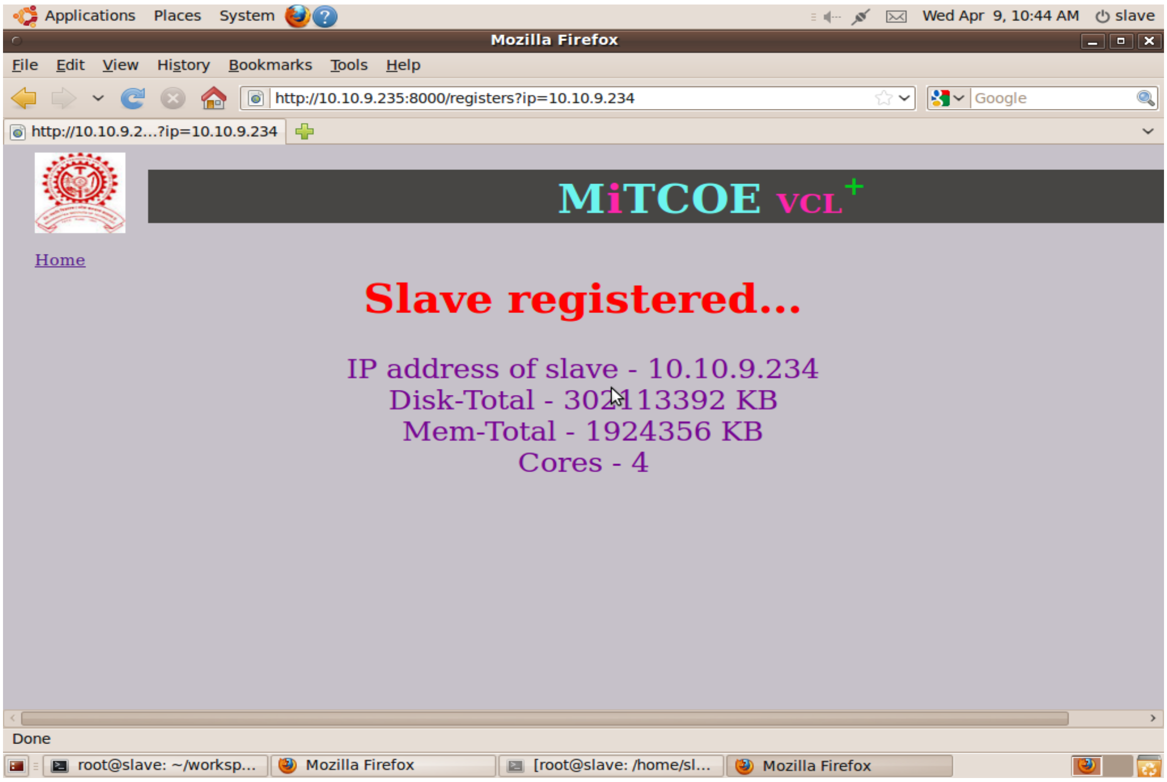

Applications Places System
Wed Apr 9, 10:39 AM slave
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://10.10.9.235:8000/register
Google
MiTCOE VCL+
Home

Enter the IP address of slave...

IP address : 10.10.9.233

Submit Query

Done
root@slave: ~/worksp...
Mozilla Firefox
root@slave: /home/sla...
Mozilla Firefox


Applications Places System
Wed Apr 9, 10:44 AM slave
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://10.10.9.235:8000/registers?ip=10.10.9.234
Google
MiTCOE VCL+
Home

Slave registered...

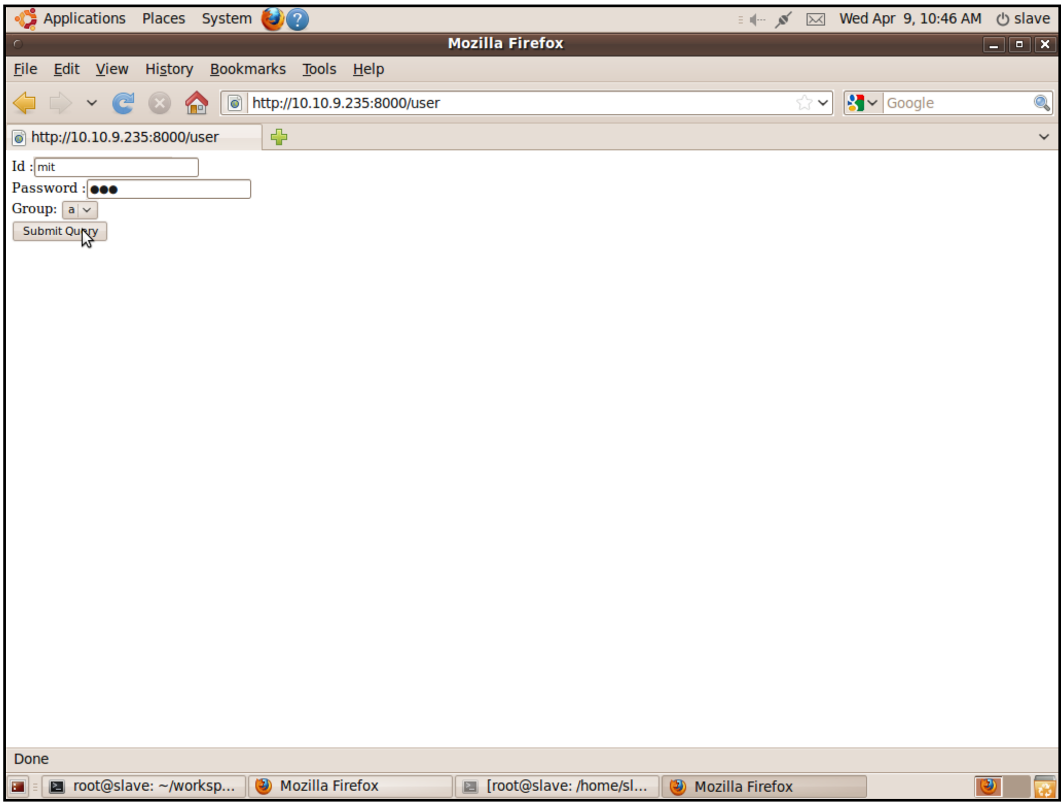
IP address of slave - 10.10.9.234
Disk-Total - 302113392 KB
Mem-Total - 1924356 KB
Cores - 4

Done
root@slave: ~/worksp...
Mozilla Firefox
[root@slave: /home/sl...
Mozilla Firefox

To Register as a User Follow the below ( Snapshot) steps:

- ➔ On “Home Screen” , Click on “Register yourself”
- ➔ Then enter “ID” and “Password” , as well select the group .

- ➔ Click on Submit Query.
- ➔ It will show a message as “User Added”



http://10.10.9.235:8000/user

Id : mit
Password : ●●●
Group: a
Submit Query

To launch the image follow the below (Snapshot) steps:

- ➔ On “Home Screen” , Click on “Log in”
- ➔ Enter the userid and password created during last steps
- ➔ Click on “ Launch image “
- ➔ Select the hours and Click on “Submit”
- ➔ As its first time launch, Select “ New VM” and click on submit
- ➔ Click on “Click here to launch the image “
- ➔ After this, it will show message as

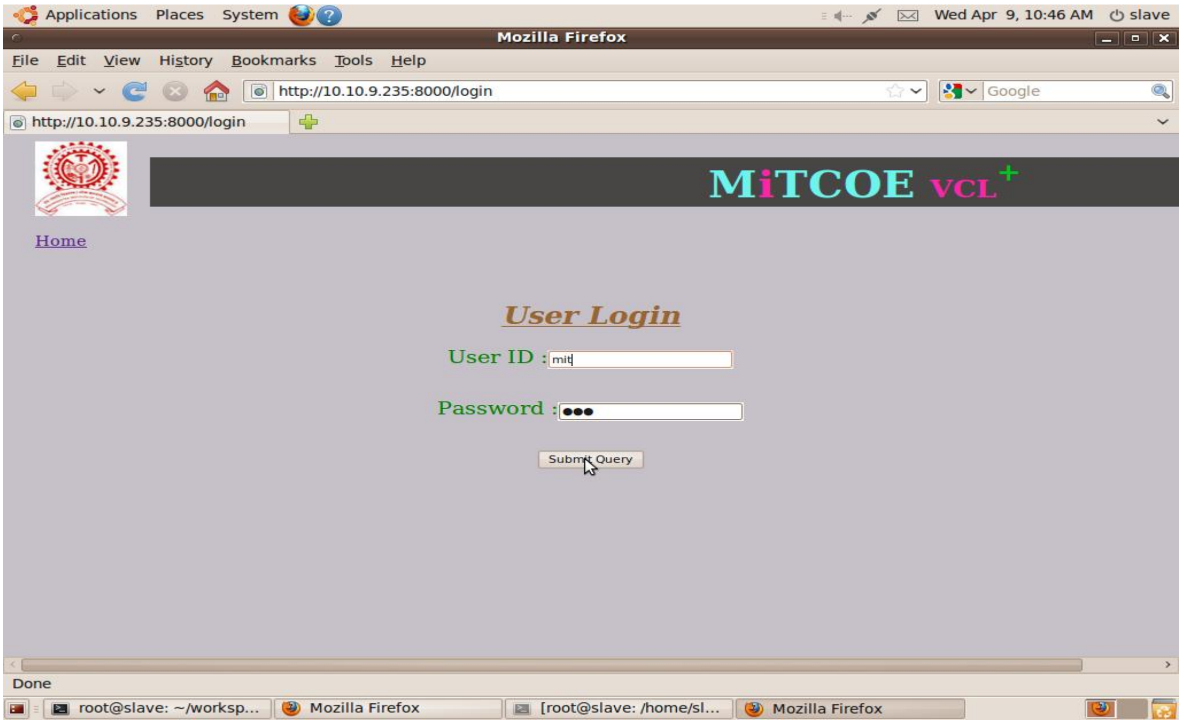
  “Kernel
  Ramdisk
  Disk”

  On the command prompt of the slave machine.
- ➔ Thus, this means that image instance has been created on the slave machine.
- ➔ Now, you can verify it by using the following command

  virsh list
- ➔ It will show the image id and its status. You can see the image by using vncviewer.

Command for it is:

```
vncviewer   0.0.0.0
```

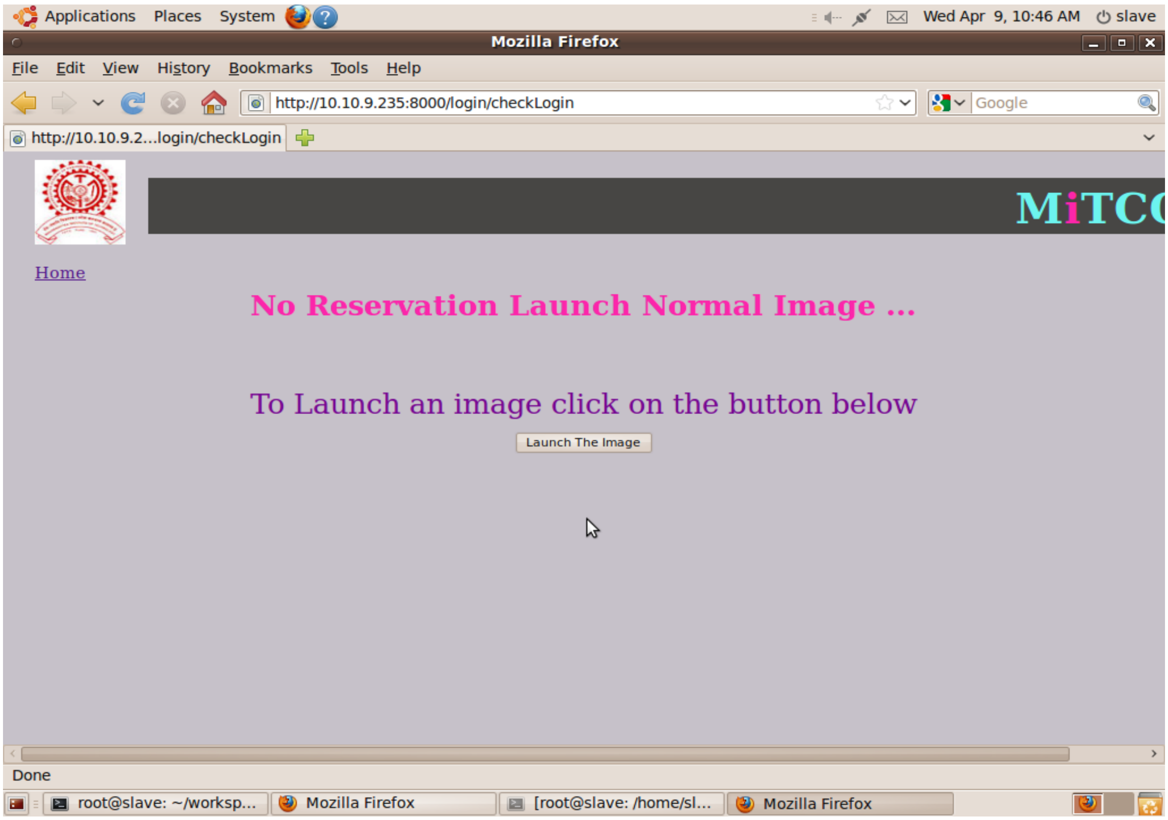


User Login

User ID : mit

Password : ●●●

Submit Query

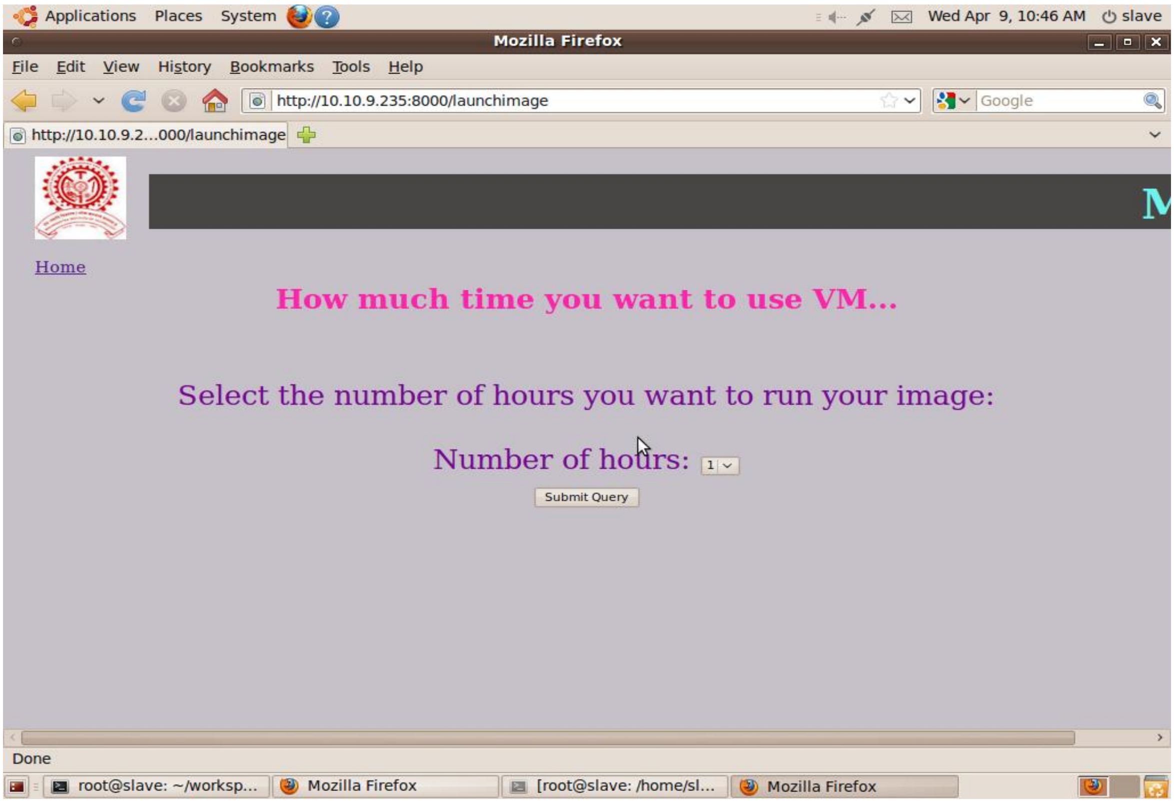


Applications Places System
Wed Apr 9, 10:46 AM
slave
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://10.10.9.235:8000/login/checkLogin
http://10.10.9.2...login/checkLogin
Google
MiTCO
Home
Done
root@slave: ~/worksp...
Mozilla Firefox
[root@slave: /home/sl...
Mozilla Firefox

No Reservation Launch Normal Image ...

To Launch an image click on the button below

Launch The Image



Applications Places System
Wed Apr 9, 10:46 AM slave
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://10.10.9.235:8000/launchimage
Google
http://10.10.9.2...000/launchimage
M
Done
root@slave: ~/worksp...
Mozilla Firefox
[root@slave: /home/sl...
Mozilla Firefox

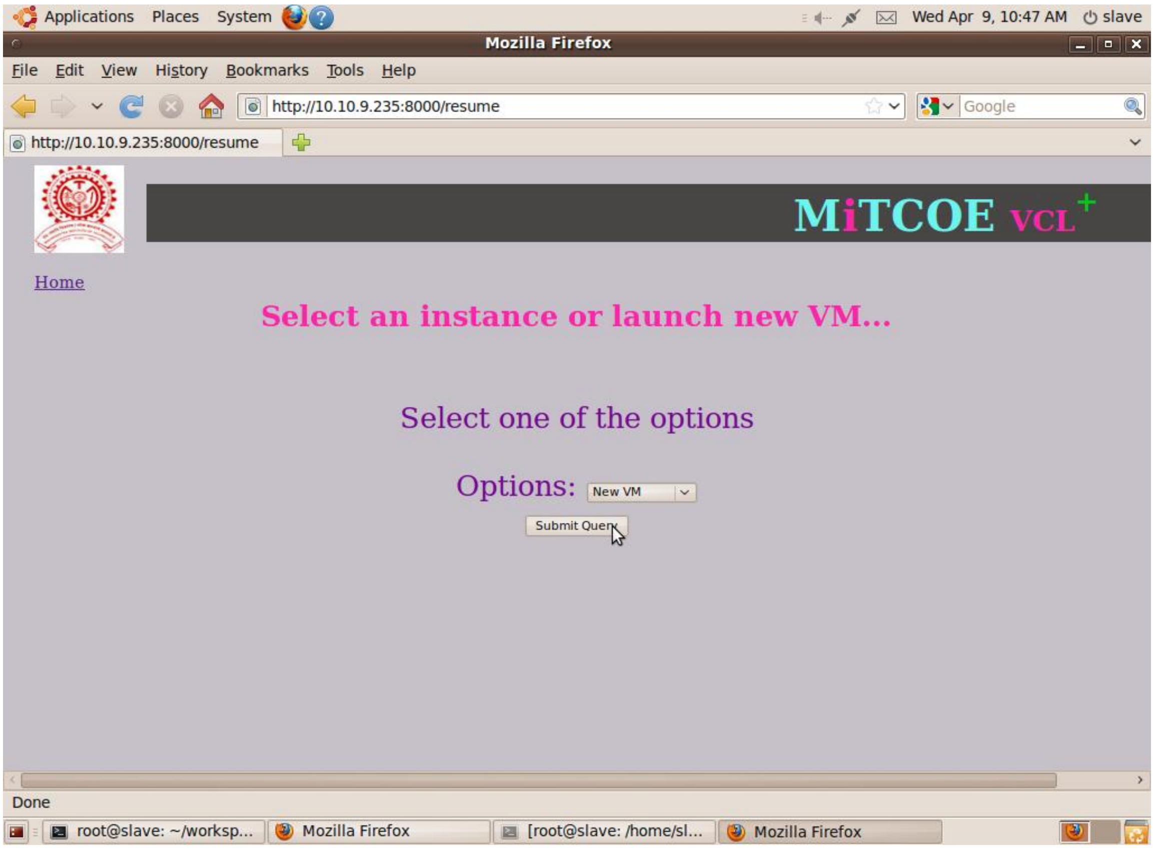
Home

How much time you want to use VM...

Select the number of hours you want to run your image:

Number of hours: 1

Submit Query



Applications Places System
Wed Apr 9, 10:47 AM slave
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://10.10.9.235:8000/resume
Google
http://10.10.9.235:8000/resume
MiTCOE VCL+
Done
root@slave: ~/worksp...
Mozilla Firefox
[root@slave: /home/sl...
Mozilla Firefox

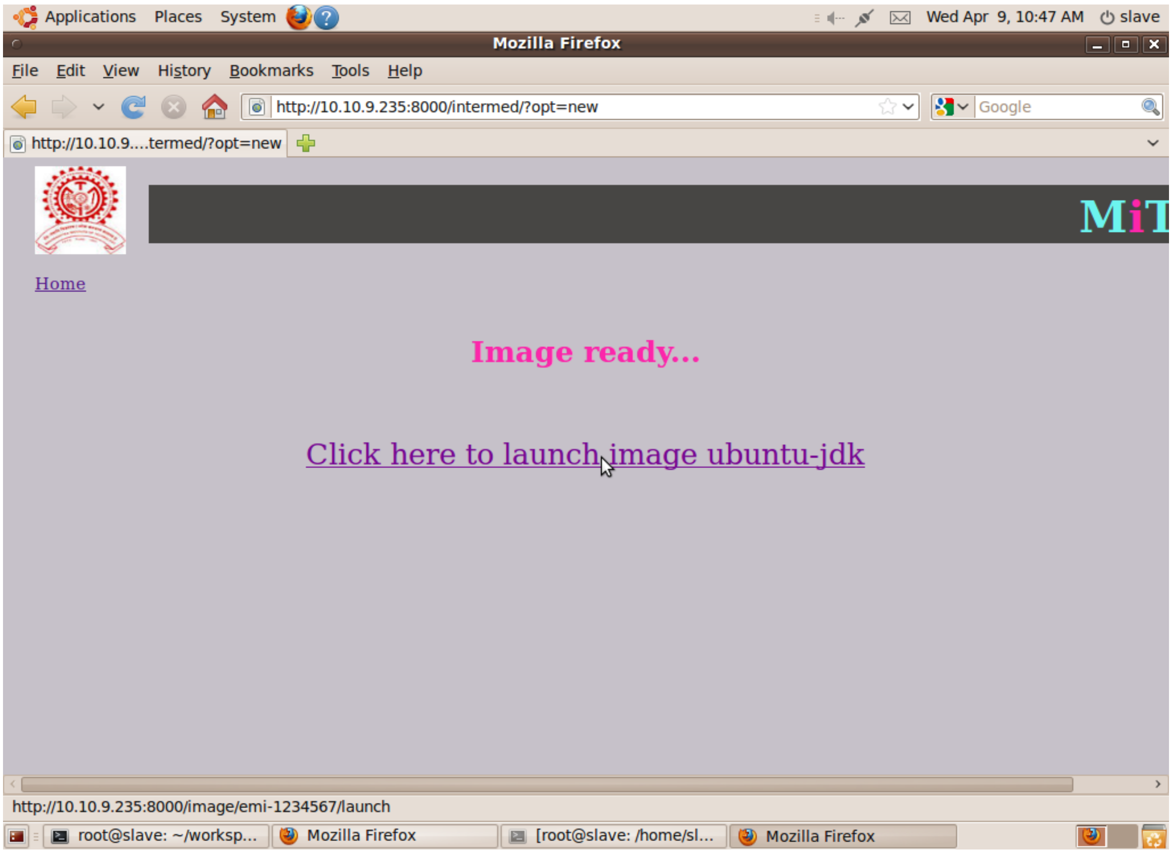
Home

Select an instance or launch new VM...

Select one of the options

Options: New VM

Submit Query



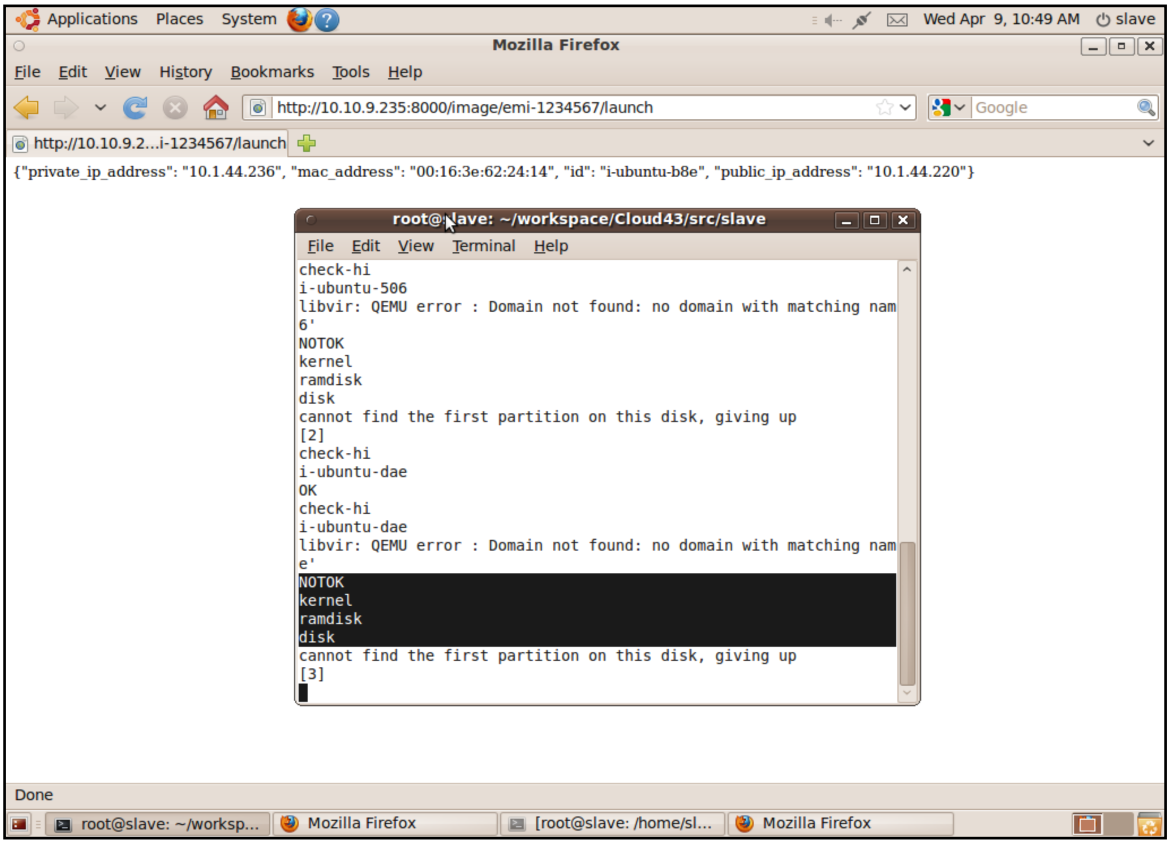
Applications Places System
Wed Apr 9, 10:47 AM slave
Mozilla Firefox
File Edit View History Bookmarks Tools Help
Google
http://10.10.9....termed/?opt=new
MiT
Home
root@slave: ~/worksp... Mozilla Firefox [root@slave: /home/sl... Mozilla Firefox

http://10.10.9.235:8000/intermed/?opt=new

Image ready...

Click here to launch image ubuntu-jdk

http://10.10.9.235:8000/image/emi-1234567/launch

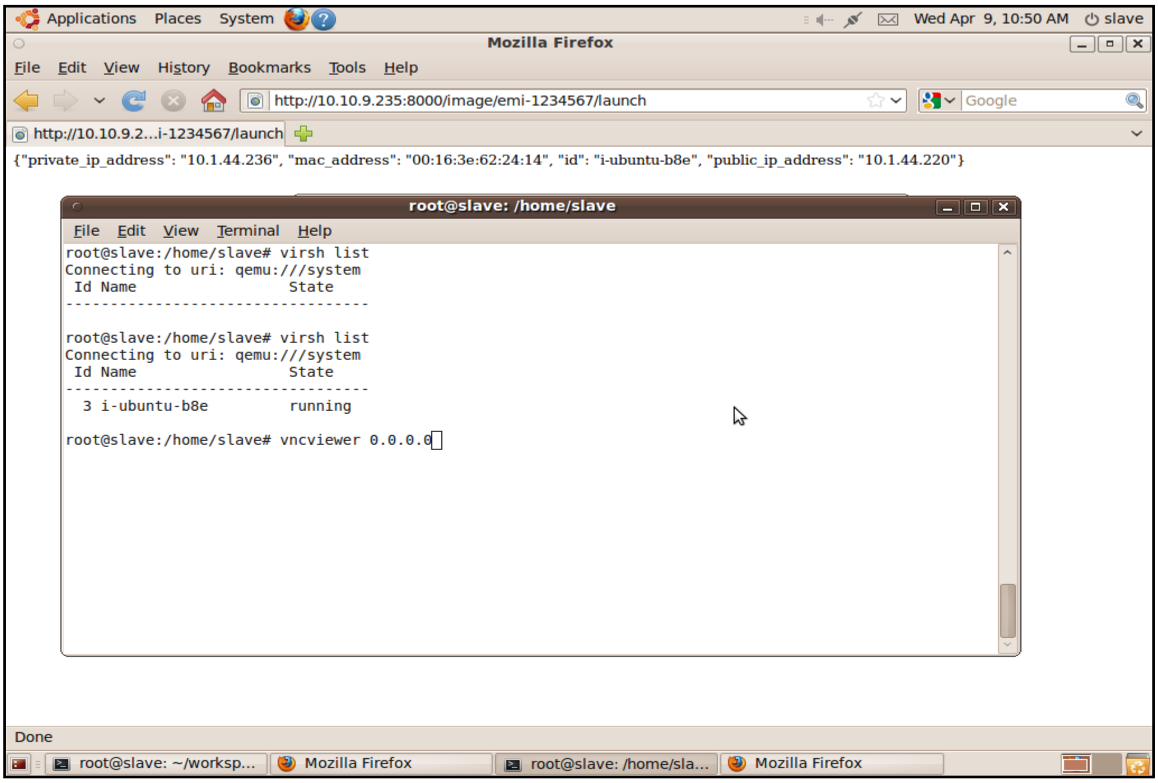


Applications Places System
Wed Apr 9, 10:49 AM slave
Mozilla Firefox
File Edit View History Bookmarks Tools Help
Google
http://10.10.9.2...i-1234567/launch
root@slave: ~/workspace/Cloud43/src/slave
File Edit View Terminal Help
Done
root@slave: ~/worksp... Mozilla Firefox [root@slave: /home/sl... Mozilla Firefox

http://10.10.9.235:8000/image/emi-1234567/launch

{"private_ip_address": "10.1.44.236", "mac_address": "00:16:3e:62:24:14", "id": "i-ubuntu-b8e", "public_ip_address": "10.1.44.220"}

```
check-hi
i-ubuntu-506
libvir: QEMU error : Domain not found: no domain with matching nam
6'
NOTOK
kernel
ramdisk
disk
cannot find the first partition on this disk, giving up
[2]
check-hi
i-ubuntu-dae
OK
check-hi
i-ubuntu-dae
libvir: QEMU error : Domain not found: no domain with matching nam
e'
NOTOK
kernel
ramdisk
disk
cannot find the first partition on this disk, giving up
[3]
```

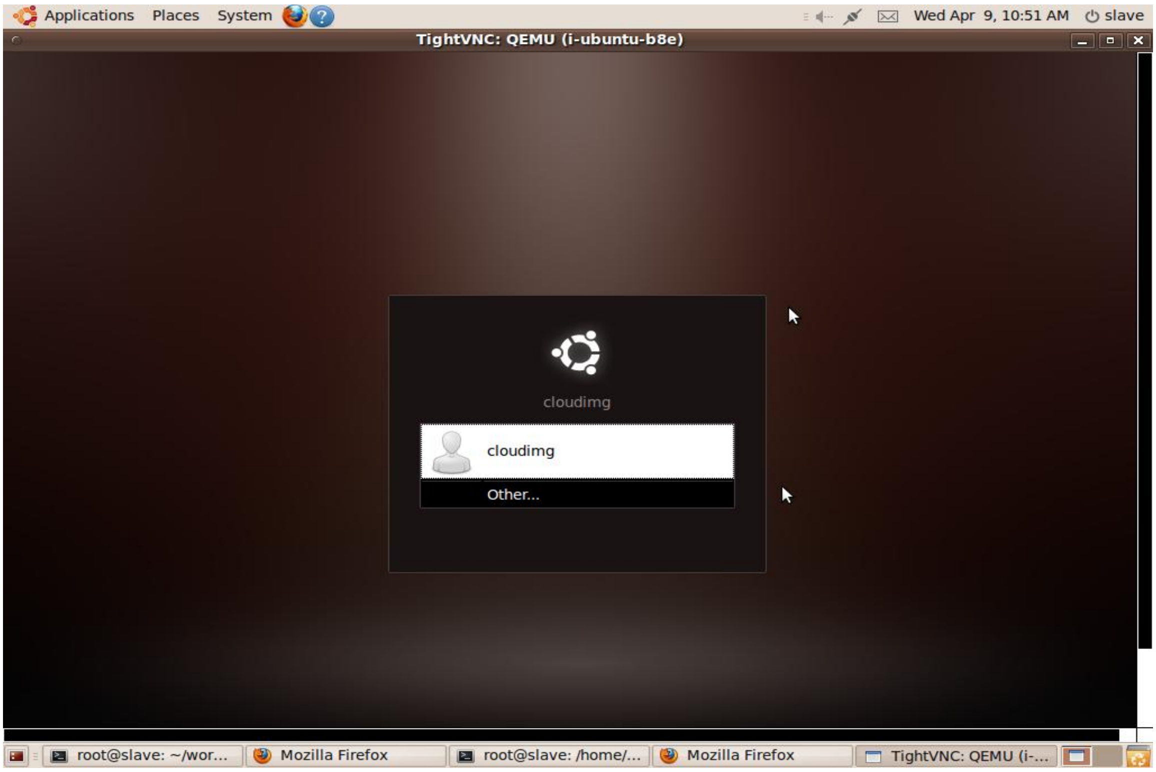

Applications Places System
Wed Apr 9, 10:50 AM slave
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://10.10.9.235:8000/image/emi-1234567/launch
Google
http://10.10.9.2...i-1234567/launch
root@slave: /home/slave
File Edit View Terminal Help
Done
root@slave: ~/worksp...
Mozilla Firefox
root@slave: /home/sla...
Mozilla Firefox

```
{"private_ip_address": "10.1.44.236", "mac_address": "00:16:3e:62:24:14", "id": "i-ubuntu-b8e", "public_ip_address": "10.1.44.220"}
```

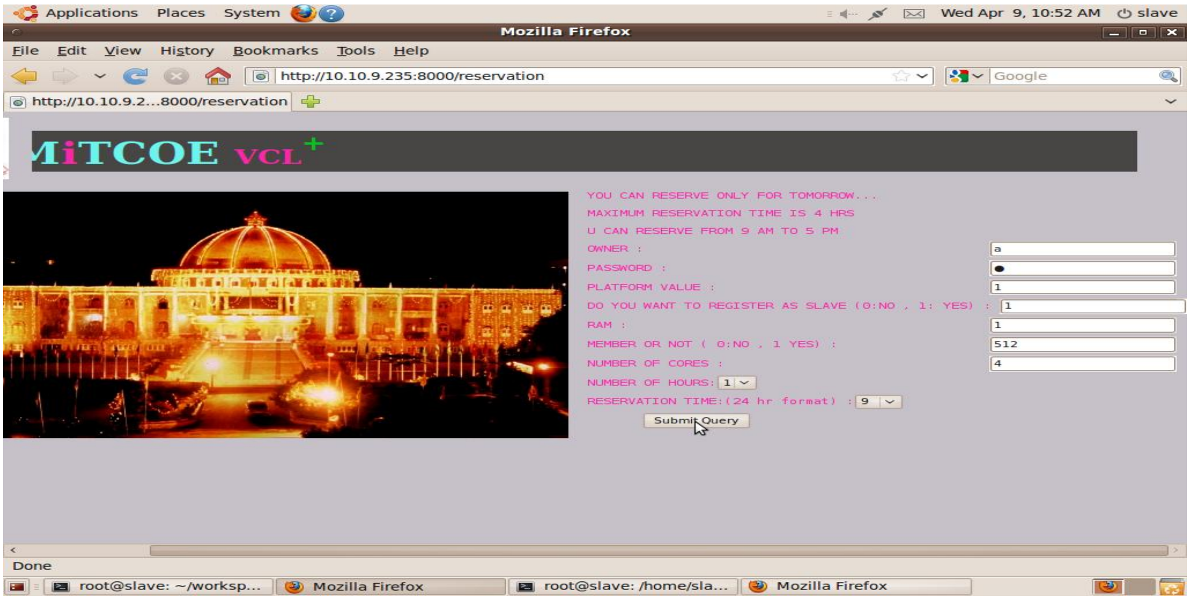
```
root@slave:/home/slave# virsh list
Connecting to uri: qemu:///system
 Id Name                 State
----------------------------------

root@slave:/home/slave# virsh list
Connecting to uri: qemu:///system
 Id Name                 State
----------------------------------
  3 i-ubuntu-b8e         running

root@slave:/home/slave# vncviewer 0.0.0.0
```



**To reserve the resources use the following snapshot:**



Applications Places System
Wed Apr 9, 10:52 AM slave
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://10.10.9.235:8000/reservation
Google
http://10.10.9.2...8000/reservation
MiTCOE VCL+
Done
root@slave: ~/worksp...
Mozilla Firefox
root@slave: /home/sla...
Mozilla Firefox

YOU CAN RESERVE ONLY FOR TOMORROW...

MAXIMUM RESERVATION TIME IS 4 HRS

U CAN RESERVE FROM 9 AM TO 5 PM

| | |
|---|---|
| OWNER : | a |
| PASSWORD : | ● |
| PLATFORM VALUE : | 1 |
| DO YOU WANT TO REGISTER AS SLAVE (0:NO , 1: YES) : | 1 |
| RAM : | 1 |
| MEMBER OR NOT ( 0:NO , 1 YES) : | 512 |
| NUMBER OF CORES : | 4 |
| NUMBER OF HOURS: | 1 |
| RESERVATION TIME:(24 hr format) : | 9 |

Submit Query

As in the above snapshot , it shows that “ ubuntu” image launched. As we know that this is the instance running on to the slave machine. So , to access it on the cline machine follow the below steps :

- ➔ First, statically configure the IP for this image
- ➔ Using the command as below , you can access it on the client machine

```
vncviewer  ip_address_of_slave_machine  : 5901
```

Example:

```
vncviewer 10.10.9.234:5901
```

# Chapter XI

# APPLICATIONS AND LIMITATIONS

## 11.1 Applications

Our architecture so produced has a lot of applications for setting VCL's
For instance when created for "College Environment"

- Cloud can be created using existing architecture therefore saving need for buying the new one
- Various virtual machine instances will be there so that the teacher can provide different instances to the students according to their need
- Own storage space for stakeholders of college so that teachers, staff, students can save their data on their respective spaces and can resume later
- Pen drives can't be detected so least possibility of cheating
- No virus problem since virus will be present for that instance only and will die as soon as that instance is closed.
- No misuse of Internet since the college can define limited bandwidth for the instances

## 11.2 Limitations

- Since Private Cloud being used Security Issues arises.
- In our adaptive cloud architecture, master does caching of virtual machine instances. When new request comes to the master, master needs to check for available resources in the pool of slaves as well as with previous client. This leads to more time required to launch an instance for a new client.
- Takes a significant amount of time to launch an instance for a new client.

# Chapter XII

# CONCLUSION AND FUTURE SCOPE

## 12.1 Conclusion

This adaptive cloud approach will improve the performance of the server and allows cloud servers to extend their computational power, thus allowing increased service capability and our Para-Scheduling algorithm will resolve all the conficts  and request between the clients. An adaptive cloud starts with one master and one slave but grows dynamically as new clients are added to a set of slaves. It ensures availability of sufficient resources.

This will be a good advancement in the way of setting up VCL for college and institutional requirements ensuring maximum efficiency with average amount of resources.

## 12.2 Future Scope

1. Automatic management for registration and allocation of computational resources to cloud users can be implemented and a work to be done to reduce time required for launching an instance for new request thus improving the performance.

2. This architecture can be successfully used rather than existing open cloud platforms thus avoiding their drawbacks and can be implemented in colleges, educational institutes to set up Virtual Computing Labs.

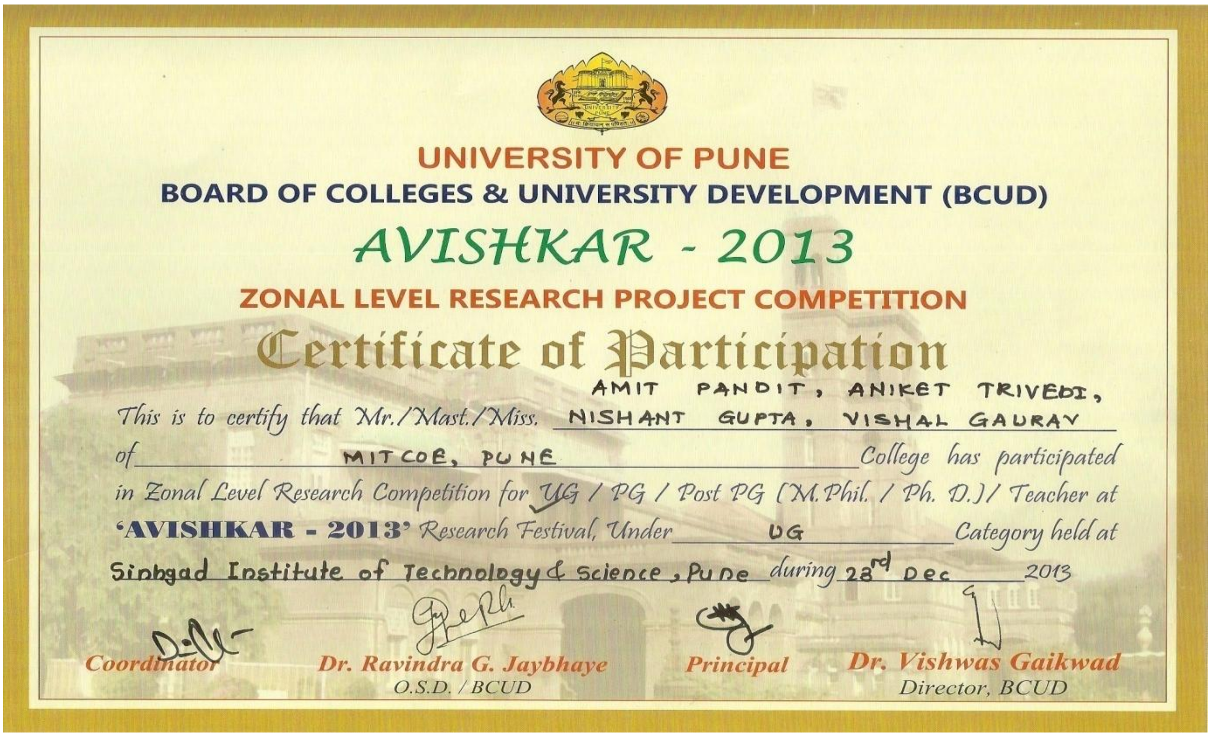
# Chapter XIII

# PARTICIPATION AND PUBLICATION

## 13.1 Project / Poster Competition

Poster Presentation at AVISHKAR – 2013
Venue: SITS, Narhe
Date of Participation: 23rd December, 2013

CERTIFICATE:



UNIVERSITY OF PUNE

BOARD OF COLLEGES & UNIVERSITY DEVELOPMENT (BCUD)

AVISHKAR - 2013

ZONAL LEVEL RESEARCH PROJECT COMPETITION

Certificate of Participation

This is to certify that Mr./Mast./Miss. AMIT PANDIT, ANIKET TRIVEDI, NISHANT GUPTA, VISHAL GAURAV of MITCOE, PUNE College has participated in Zonal Level Research Competition for [✓ UG] / PG / Post PG (M.Phil. / Ph. D.)/ Teacher at 'AVISHKAR - 2013' Research Festival, Under UG Category held at Sinhgad Institute of Technology & Science, Pune during 28rd Dec 2013

Coordinator

Dr. Ravindra G. Jaybhaye
O.S.D. / BCUD

Principal

Dr. Vishwas Gaikwad
Director, BCUD

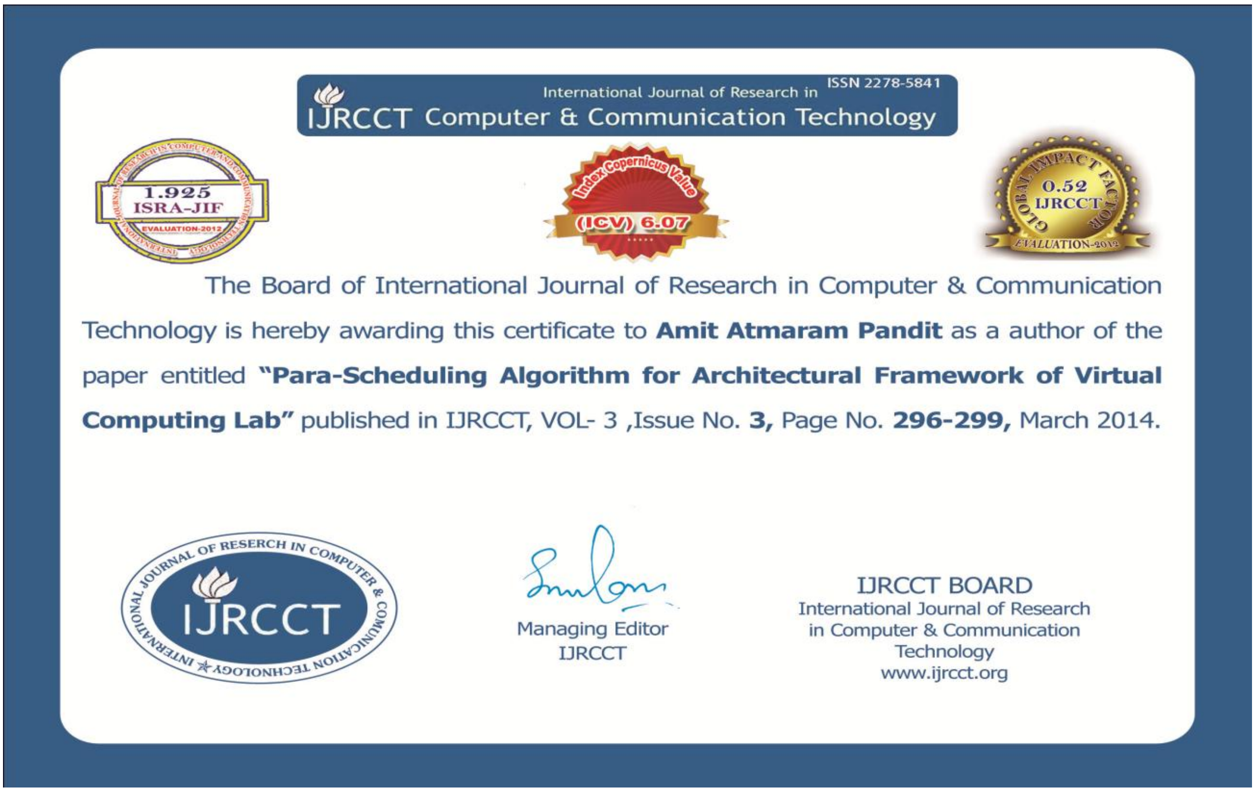
## 13.2 Research Paper Publication

**Journal name**: International Journal of Research in Computer and Communication Technology, Vol 3, Issue 3, March- 2014 [ISSN (O) 2278-5841]

**Impact Factor:** 1.92

**Online Paper Link**: http://www.ijrcct.org/index.php/ojs/article/view/597

CERTIFICATE:



IJRCCT International Journal of Research in Computer & Communication Technology ISSN 2278-5841

1.925 ISRA-JIF EVALUATION-2012

Index Copernicus Value (ICV) 6.07

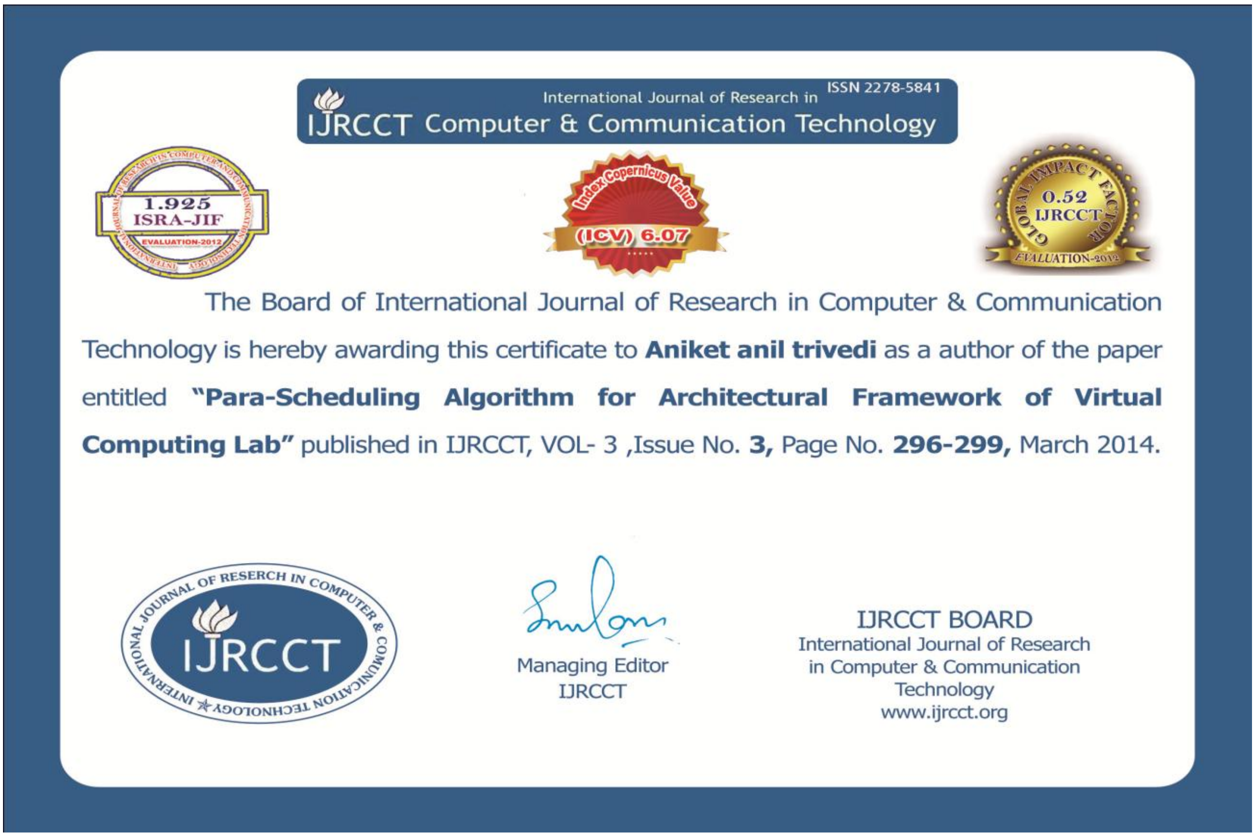
GLOBAL IMPACT FACTOR 0.52 IJRCCT EVALUATION-2012

The Board of International Journal of Research in Computer & Communication Technology is hereby awarding this certificate to **Amit Atmaram Pandit** as a author of the paper entitled **"Para-Scheduling Algorithm for Architectural Framework of Virtual Computing Lab"** published in IJRCCT, VOL- 3 ,Issue No. **3,** Page No. **296-299,** March 2014.

INTERNATIONAL JOURNAL OF RESERCH IN COMPUTER & COMUNICATION TECHNOLOGY IJRCCT

Managing Editor
IJRCCT

IJRCCT BOARD
International Journal of Research
in Computer & Communication
Technology
www.ijrcct.org



IJRCCT International Journal of Research in Computer & Communication Technology ISSN 2278-5841

1.925 ISRA-JIF EVALUATION-2012

Index Copernicus Value (ICV) 6.07

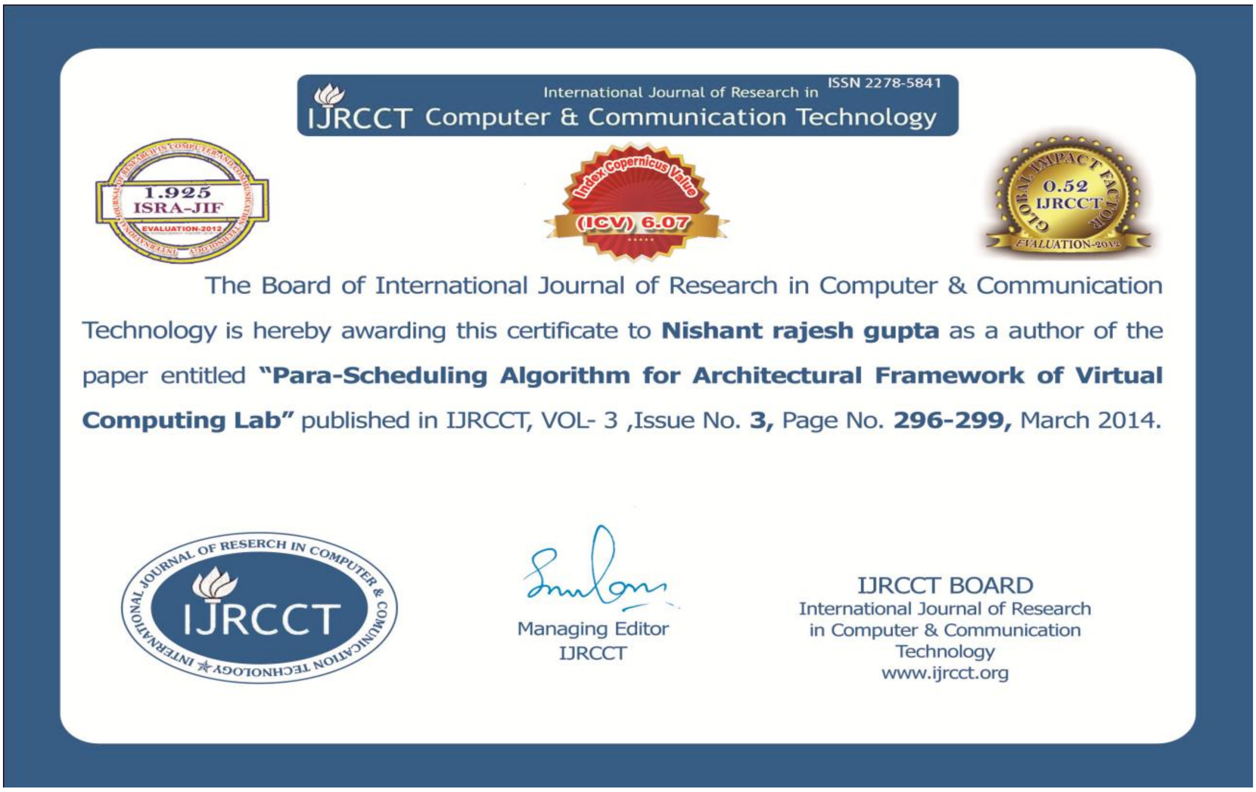
GLOBAL IMPACT FACTOR 0.52 IJRCCT EVALUATION-2012

The Board of International Journal of Research in Computer & Communication Technology is hereby awarding this certificate to **Aniket anil trivedi** as a author of the paper entitled **"Para-Scheduling Algorithm for Architectural Framework of Virtual Computing Lab"** published in IJRCCT, VOL- 3 ,Issue No. **3,** Page No. **296-299,** March 2014.

INTERNATIONAL JOURNAL OF RESERCH IN COMPUTER & COMUNICATION TECHNOLOGY IJRCCT

Managing Editor
IJRCCT

IJRCCT BOARD
International Journal of Research
in Computer & Communication
Technology
www.ijrcct.org



IJRCCT International Journal of Research in Computer & Communication Technology ISSN 2278-5841

1.925 ISRA-JIF EVALUATION-2012

Index Copernicus Value (ICV) 6.07

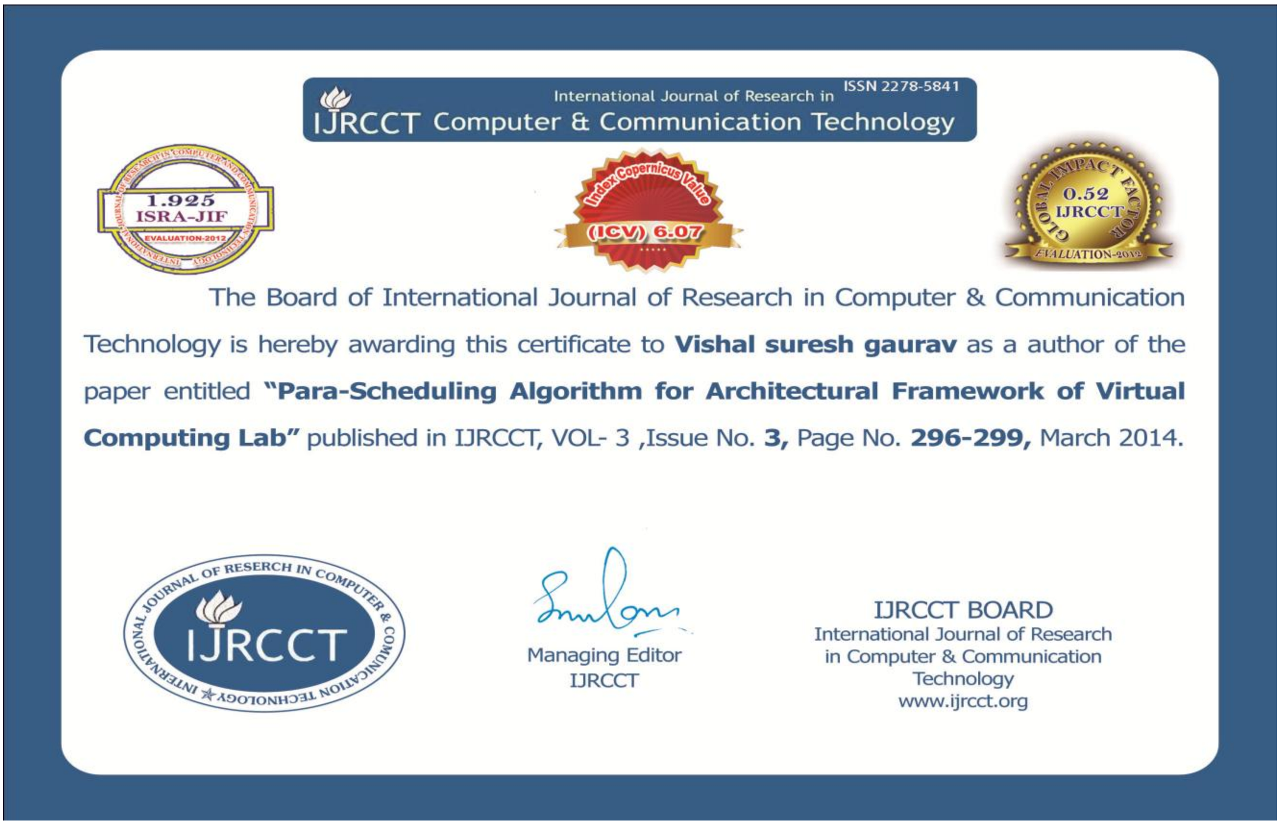
GLOBAL IMPACT FACTOR 0.52 IJRCCT EVALUATION-2012

The Board of International Journal of Research in Computer & Communication Technology is hereby awarding this certificate to **Nishant rajesh gupta** as a author of the paper entitled **"Para-Scheduling Algorithm for Architectural Framework of Virtual Computing Lab"** published in IJRCCT, VOL- 3 ,Issue No. **3,** Page No. **296-299,** March 2014.

INTERNATIONAL JOURNAL OF RESERCH IN COMPUTER & COMUNICATION TECHNOLOGY ★ IJRCCT

Managing Editor
IJRCCT

IJRCCT BOARD
International Journal of Research
in Computer & Communication
Technology
www.ijrcct.org



IJRCCT International Journal of Research in Computer & Communication Technology ISSN 2278-5841

1.925 ISRA-JIF EVALUATION-2012

Index Copernicus Value (ICV) 6.07

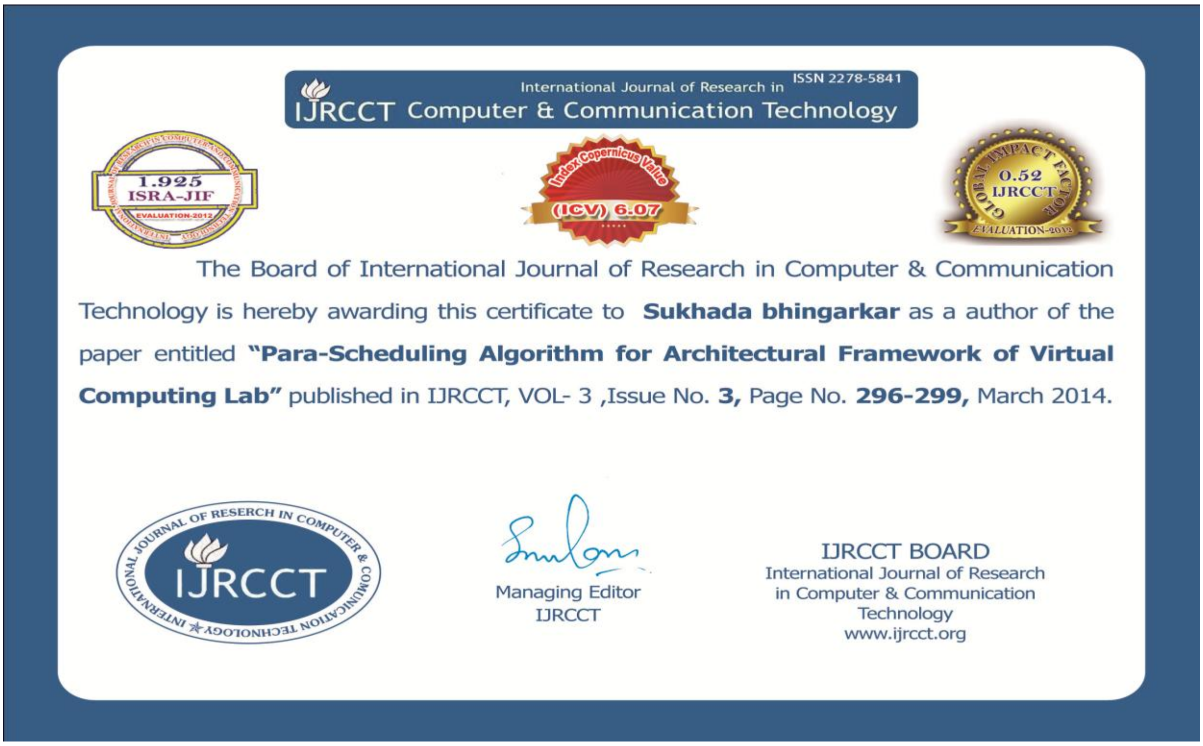
GLOBAL IMPACT FACTOR 0.52 IJRCCT EVALUATION-2012

The Board of International Journal of Research in Computer & Communication Technology is hereby awarding this certificate to **Vishal suresh gaurav** as a author of the paper entitled **"Para-Scheduling Algorithm for Architectural Framework of Virtual Computing Lab"** published in IJRCCT, VOL- 3 ,Issue No. **3,** Page No. **296-299,** March 2014.

INTERNATIONAL JOURNAL OF RESERCH IN COMPUTER & COMUNICATION TECHNOLOGY ★ IJRCCT

Managing Editor
IJRCCT

IJRCCT BOARD
International Journal of Research
in Computer & Communication
Technology
www.ijrcct.org



IJRCCT International Journal of Research in Computer & Communication Technology ISSN 2278-5841

1.925 ISRA-JIF EVALUATION-2012

Index Copernicus Value (ICV) 6.07

GLOBAL IMPACT FACTOR 0.52 IJRCCT EVALUATION-2012

The Board of International Journal of Research in Computer & Communication Technology is hereby awarding this certificate to **Sukhada bhingarkar** as a author of the paper entitled **“Para-Scheduling Algorithm for Architectural Framework of Virtual Computing Lab”** published in IJRCCT, VOL- 3 ,Issue No. **3,** Page No. **296-299,** March 2014.

INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER & COMUNICATION TECHNOLOGY IJRCCT

Managing Editor
IJRCCT

IJRCCT BOARD
International Journal of Research
in Computer & Communication
Technology
www.ijrcct.org

# ANNEXURE A

# PROJECT ANALYSIS OF ALGORITHM DESIGN

## 1. Mathematical modeling

Mathematically the problem can be expressed as follows,

Let M be a master with finite set of resources defined by following tuples

M = ( c , m , s , b )

Where,

C = CPU resources

M = memory resources

S = disk storage

B = network bandwidth

Let A = ( a1 , a2 , a3 , -------------, an ) be a set of clients , and let S = ( s1 , s2 , s3 , --------------, sn ) be a set of slaves , where each slave

$$Sk = ( Sk^{CPU} , Sk^{RAM} )$$

Specifies the CPU capacity and the memory capacity of the Virtual Machine.

When client ai makes a request or VM instance to the master M,

M checks the priority of request s and then the request with maximum priority is forwarded to a slave$Sk$.

$Sk$ Allocates necessary resources for launching requested VM instance.

After running VM instance, if ai wishes then it is registered as a slave into set S.

Hence S is updated as,

$$S = \sum Sk \cup ai$$

Calculation OF Priority

$$PF = PfV + RAS - \log_2 R - \ln E + MON$$

Where,

$PfV$ = Platform value

$RAS$ = ReqAs a Slave

R = RAM required in MB

E = Execution time in minute

$MON$ = Member Or Not

Actual Priority

$$Pi = 10 + \ 0.20PF$$

Based upon the $Pi$ value master will decide the scheduling of request

If two requests have same $Pi$value, then

i) Check which of them is ready to register as slave, one with positive value is scheduled first .

ii) If both have same value of $RAS$ , then schedule it on first come first serve basis.

**Algorithm**

1. Client sends a request to server with the required information.
2. Server have list of requests ri of client ai .
3. A) i) If there's only one request then server checks the availability of resources.

   ii) If the resource is available then a connection is established between current client and the slave.

   B) i) If the resource is not available then the server checks the priority of a client using the resource and the client demanding the resource.

   ii) One having greater priority get access to resource.

   iii) If both have same priority then the demanding client has to wait until the resource is free.

4. I) If there are multiple requests , then

   ii) First of all priority of the entire request is calculated.

   iii) If two or more requests have same priority then they are scheduled on first come first serve basis.

iv) Next step is checking the availability of resources which is same as for one request.

5. After permission checking , availability of requested resource is checked
   i) If the resource is available then it is assigned to the client
   ii) Else given resource is installed and then assigned to the client.

6. When clients request is fulfilled, the resource is released and database is updated.
7. Now if the client wants to register itself as a slave it is registered and S is updated as

$$S = \sum Sk \cup ai$$

8. While there are unprocessed request go to step 3.
9. End

## 2. Function point analysis

Function points play a significant role in the management of information systems. Function point analysis is a proven, reliable method for measuring application development work-products.

One of the initial design criteria for function points is to provide a mechanism that both software developers and users can utilize to define functional requirements. It is determined that the best way to gain an understanding of the users' needs is to approach their problem from the perspective of how they view the results an automated system produces. Therefore, one of the primary goals of Function Point Analysis is to evaluate a system's capabilities from a user's point of view. To achieve this goal, the analysis is based upon the various ways users interact with computerized systems. From a user's perspective a system assists them in doing their job by providing five (5) basic functions. Two of these address the data requirements of an end user and are referred to as Data Functions. The remaining three address the user's need to access data and are referred to as Transactional Functions.

**The Five Components of Function Points**

► **Data Functions**

- Internal Logical Files
- External Interface Files

► **Transactional Functions**

- External Inputs
- External Outputs
- External Inquiries

1. **Internal Logical Files** (ILF)

The first data function allows users to utilize data they are responsible for maintaining.

In our software data entered by end users are the platform they want to use , RAM size , CPU requirement and TIME of execution.

Table 9.1 ILF

| **Platform To Use** | **RAM Size** | **CPU Type** | **Time Of Execution** | **RAS** |
|---|---|---|---|---|
| Eclipse | 600 mb | | 45 minutes | yes |
| DOS Box | 200 mb | | 120 minutes | no |

2. **External Interface Files**

The second Data Function a system provides an end user is also related to logical groupings of data. In this case the user is not responsible for maintaining the data. The data resides in another system and is maintained by another user or system. The user of the system being counted requires this data for reference purposes only.

In our software these files are the database stored by Master PC and are CPU resources , Memory Resources , Disk storage , Network Bandwidth.

Table 9.2 EIF

| **CPU Resources** | **Memory Resources** | **Disk Storage** | **Network Bandwidth** |
|---|---|---|---|
| | | | |

3. **External Input**

 The first Transactional Function allows a client to maintain Internal Logical Files (ILFs) through the ability to add, change and delete the data. For example, a client can change his time of execution. In this case the client is utilizing a transaction referred to as an External Input (EI). An External Input gives the client the capability to maintain the data in ILF's through adding, changing and deleting its contents.

4. **External Output**

The next Transactional Function gives the MASTER the ability to produce outputs. For example a pilot has the ability to separately display ground speed, true air speed and calibrated air speed. The results displayed are derived using data that is maintained and data that is referenced. In function point terminology the resulting display is called an External Output (EO).

5. **External Inquiries**

The final capability provided to client through a computerized system addresses the requirement to select and display specific data from files. To accomplish this a client inputs selection information that is used to retrieve data that meets the specific criteria. In this situation there is no manipulation of the data. It is a direct retrieval of information contained on the files. For example if the MASTER displays availability of resources then the client can use those resources. These transactions are referred to as External Inquiries (EQ).

# REFERENCES AND GLOSSARY

## 9.1 References

[1] Prof. Sukhada Bhingarkar, Prof. Bharati Ainapure and Prof. Devan Shah ”A New Adaptive Private Cloud Architecture”

[2] Prof. Sukhada Bhingarkar, Prof. Bharati Ainapure and Prof. Devan Shah ” Architectural Framework for Diffused Cloud”

[3] Prof. Sukhada Bhingarkar, Prof. Bharati Ainapure and Prof. Devan Shah”A Novel Architecture Style: Diffused Cloud for Virtual Computing Lab”

[4] Peter Sempolinski and Douglas Thain, University of Notre Dame “ A Comparison and Critique of Eucalyptus, OpenNebula and Nimbus”

[5] Cloud computing and its services:

http://www.webopedia.com/TERM/C/cloud_computing.html

[6] Services provided by cloud:

http://bitheads.com/what-types-of-cloud-services-are-available/

[7] Virtualization and virtual computing:

http://www.vmware.com/in/virtualization/

[8] Open Nebula and its technology:

http://opennebula.org/about:technology

[9] Open Stack and its technology :

http://www.ubuntu.com/cloud/tools/openstack/reference-architecture

[10] Web source for eucalyptus: Eucalyptus.com

## 9.2 Glossary

CRM : Customer Relationship Management

EC2 : Elastic Cloud Compute

MAC : Media Access Control

VM : Virtual Machine

VT : Virtual Technology

NIC : Network Interface Card

NFS : New file system