PVG's
COLLEGE OF ENGINEERING
Nashik

Department of Computer Engineering

LABORATORY MANUAL

2018-2019

LABORATORY PRACTICE-IV

[Cloud Computing]

BE-COMPUTER ENGINEERING

SEMESTER-II

Subject Code: **410255**

TEACHING SCHEME

Practical: 4 Hrs/Week

EXAMINATION SCHEME

Oral: 50 Marks

Term Work: 50 Marks

-: Name of Faculty:-

Prof. A.R.Jain

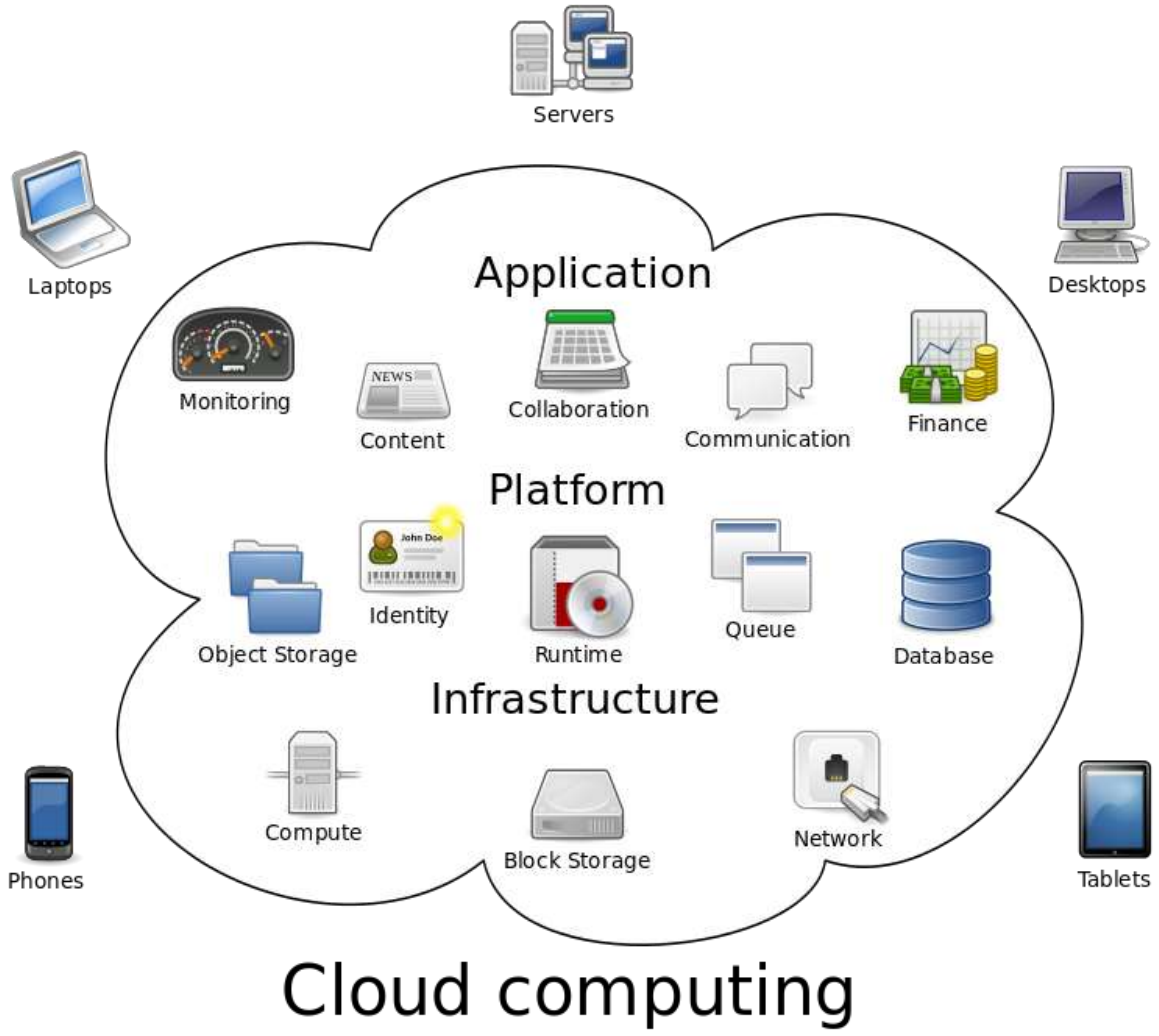
Asst. Professor, Department of Computer Engineering.

Title: Installation and configuration of own Cloud

Theory:

What is Cloud Computing?

Cloud computing is a method for delivering information technology (IT) services in which resources are retrieved from the Internet through web-based tools and applications, as opposed to a direct connection to a server. Rather than keeping files on a proprietary hard drive or local storage device, cloud-based storage makes it possible to save them to a remote database. As long as an electronic device has access to the web, it has access to the data and the software programs to run it.



Cloud Computing – Types of Cloud

Cloud computing is usually described in one of two ways. Either based on the deployment model, or on the service that the cloud is offering.

Based on a deployment model, we can classify cloud as

- **Public,**
- **Private**
- **Hybrid**
- **Community cloud**

Based on a service the cloud model is offering, we are speaking of either:

- **IaaS (Infrastructure-as-a-Service)**
- **PaaS (Platform-as-a-Service)**
- **SaaS (Software-as-a-Service)**
- **or, Storage, Database, Information, Process, Application, Integration, Security, Management, Testing-as-a-service**

Basically, programs that are needed to run a certain application are now more popularly located on a remote machine, owned by another company. This is done in order not to lose on the quality performance due to processing power of your own computer, to save money on IT support, and yet remain advantageous on the market. These computers that run the applications, store the data, and use a server system, are basically what we call "the cloud".

Public Cloud

When we talk about **public cloud**, we mean that the whole computing infrastructure is located on the premises of a cloud computing company that offers the cloud service. The location remains, thus, separate from the customer and he has no physical control over the infrastructure.

As public clouds use shared resources, they do excel mostly in performance, but are also most vulnerable to various attacks.

GlobalDots offers worldwide Public Cloud service in leading data centers. Our experts will assist you in choosing the right solution for you.

Private Cloud

Private Cloud provides the same benefits of Public Cloud, but uses dedicated, private hardware. Private cloud means using a cloud infrastructure (network) solely by one customer/organization. It is not shared with others, yet it is remotely located. The companies have an option of choosing an on-premise private cloud as well, which is more expensive, but they do have a physical control over the infrastructure.

The security and control level is highest while using a private network. Yet, the cost reduction can be minimal, if the company needs to invest in an on-premise cloud infrastructure.

GlobalDots offers worldwide private cloud service in leading data centers.

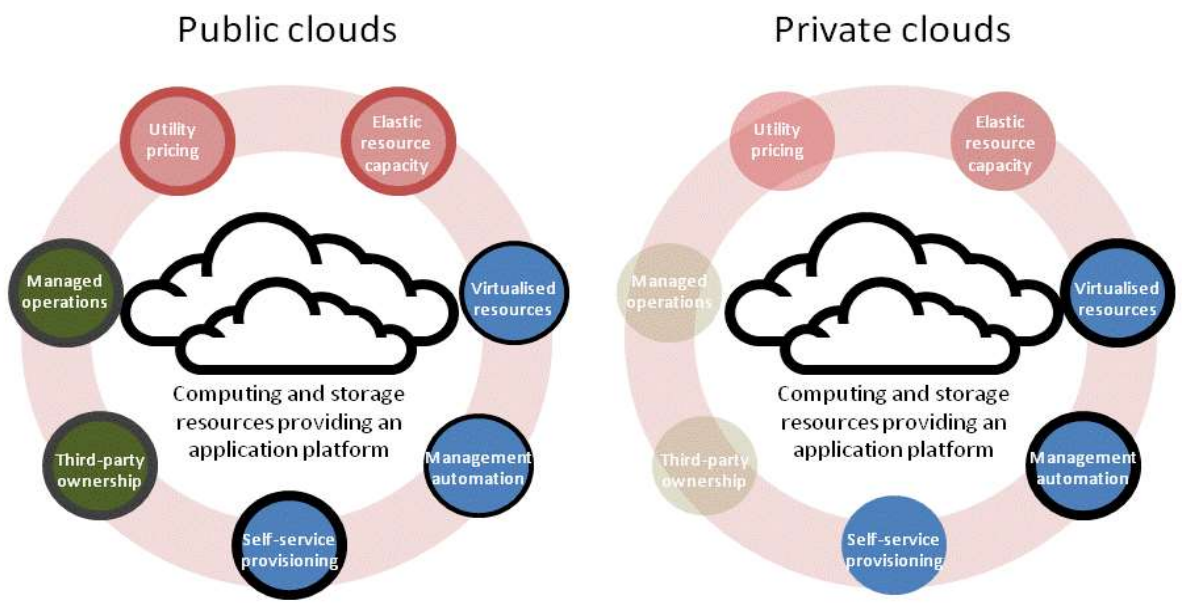
With our Private Cloud you'll get:

- Increased redundancy
- Decreased provisioning time for new servers
- Saved capital by eliminating hardware support contracts
- Quicker expendability compared to hosting your own physical servers
- Use of dedicated, private hardware

Hybrid Cloud

Hybrid cloud, of course, means, using both private and public clouds, depending on their purpose. For example, public cloud can be used to interact with customers, while keeping their data secured through a private cloud. Most people associate traditional public cloud service with elastic scalability and the ability to handle constant shifts in demand. However, performance issues can arise for certain data-intensive or high-availability workloads.

GlobalDots offer combines hybrid cloud with bare-metal and virtualized clouds into a unified environment allowing your business to optimize for scale performance and cost simultaneousl



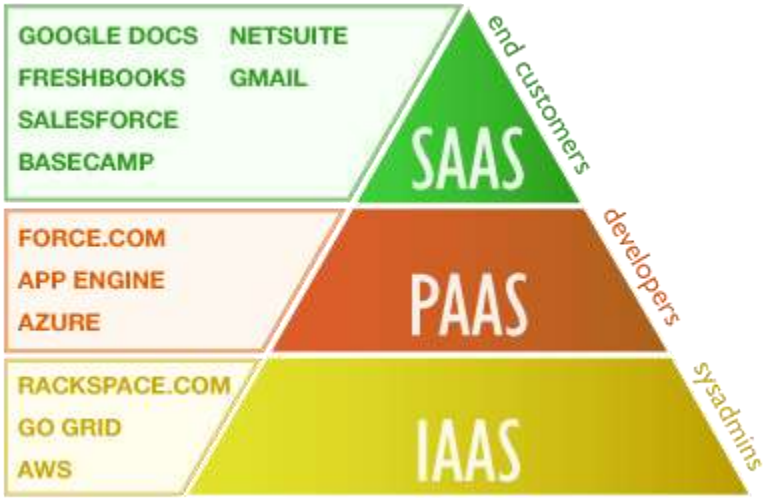
Community cloud

It implies an infrastructure that is shared between organizations, usually with the shared data and data management concerns. For example, a community cloud can belong to a government of a single country. Community clouds can be located both on and off the premises.

The most popular services of the cloud are that of either **infrastructure**, **platform**, **software**, or **storage**.

As explained before, the most common cloud service is that one offering data storage disks and virtual servers, i.e. infrastructure. Examples of Infrastructure-as-a-Service (IaaS) companies are Amazon, Rackspace, Flexiscale.

If the cloud offers a development platform, and this includes operating system, programming language execution environment, database, and web server, the model is known as Platform-as-a-Service (PaaS), examples of which are Google App Engine, Microsoft Azure, Salesforce. Operating system can be frequently upgraded and developed with PaaS, services can be obtained from diverse sources, and programming can be worked in teams (geographically distributed).
Software-as-a-Service (SaaS), finally, means that users can access various software applications on a pay-per-use basis. As opposed to buying licensed programs, often very expensive. Examples of such services include widely used GMail, or Google Docs.



Advantages of Cloud Computing:

1. Less Costs

The services are free from capital expenditure. There are no huge costs of hardware in cloud computing. You just have to pay as you operate it and enjoy the model based on your subscription plan.

2. 24 X 7 Availability

Most of the cloud providers are truly reliable in offering their services, with most of them maintaining an uptime of 99.9%. The workers can get onto the applications needed basically from anywhere. Some of the applications even function off-line.

3. Flexibility in Capacity

It offers flexible facility which could be turned off, up or down as per the circumstances of the user. For instance, a promotion of sales is very popular, capacity can be immediately and quickly added to it for the avoidance of losing sales and crashing servers. When those sales are done, the capacity can also be shrunk for the reduction of costs.

4. All over Functioning

Cloud computing offers yet another advantage of working from anywhere across the globe, as long as you have an internet connection. Even while using the critical cloud services that offer mobile apps, there is no limitation of the device used.

5. Automated Updates on Software

In cloud computing, the server suppliers regularly update your software including the updates on security, so that you do not need to agonize on wasting your crucial time on maintaining the system. You find extra time to focus on the important things like ‘How to grow your businesses.

6. Security

Cloud computing offers great security when any sensitive data has been lost. As the data is stored in the system, it can be easily accessed even if something happens to your computer. You can even remotely wipe out data from the lost machines for avoiding it getting in the wrong hands.

7. Carbon Footprint

Cloud computing is helping out organizations to reduce their carbon footprint. Organizations utilize only the amount of resources they need, which helps them to avoid any over-provisioning. Hence, no waste of resources and thus energy.

8. Enhanced Collaboration

Cloud applications enhance collaboration by authorizing diverse groups of people virtually meet and exchange information with the help of shared storage. Such capability helps in improving the customer service and product development and also reducing the marketing time.

9. Control on the Documents

Before cloud came into being, workers needed to send files in and out as the email attachments for being worked on by a single user at one time ultimately ending up with a mess of contrary titles, formats, and file content. Moving to cloud computing has facilitated central file storage.

10. Easily Manageable

Cloud computing offers simplified and enhanced IT maintenance and management capacities by agreements backed by SLA, central resource administration and managed infrastructure. You get to enjoy a basic user interface without any requirement for installation. Plus you are assured guaranteed and timely management, maintenance, and delivery of the IT services.

Applications of Cloud Computing

1. Online File storage
2. Photo editing software
3. Digital video software
4. Twitter-related applications
5. Creating image-album
6. Web application for antivirus
7. Word processing application
8. Spreadsheets
9. Presentation software
10. Finding a way on the map
11. E-commerce software
12. Miscellaneous applications.

Conclusion:

Title:

Implementation of Virtualization in Cloud Computing to Learn Virtualization Basics, Benefits of Virtualization in Cloud using Open Source Operating System.

Theory:

What is Virtualization in Cloud Computing?

Virtualization is the "creation of a virtual (rather than actual) version of something, such as a server, a desktop, a storage device, an operating system or network resources". In other words, Virtualization is a technique, which allows sharing a single physical instance of a resource or an application among multiple customers and organizations. It does by assigning a logical name to a physical storage and providing a pointer to that physical resource when demanded.

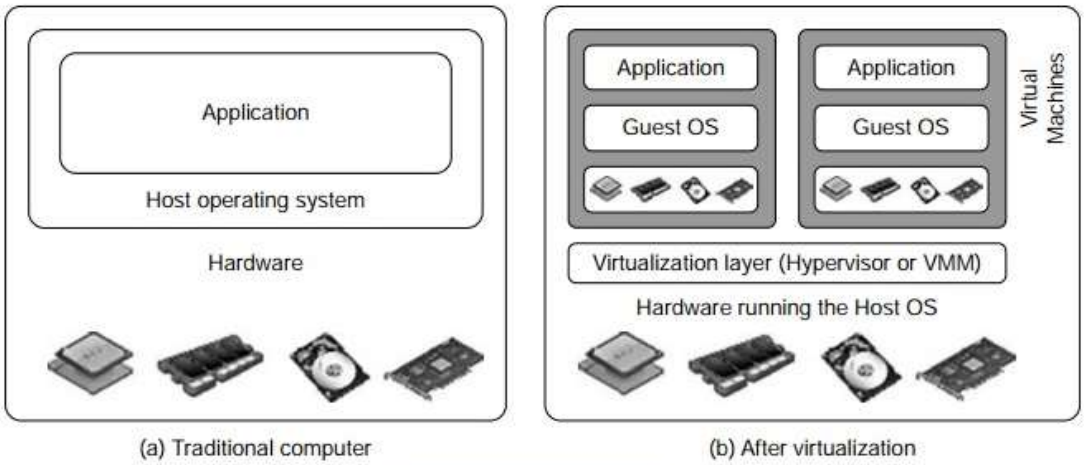


Fig: Traditional Computer Vs Virtualization

Types of Virtualization:

1. Hardware Virtualization.
2. Operating system Virtualization.
3. Server Virtualization.

4. Storage Virtualization.

1) Hardware Virtualization:

When the virtual machine software or virtual machine manager *(VMM) is directly installed on the hardware system* is known as hardware virtualization. The main job of hypervisor is to control and monitoring the processor, memory and other hardware resources. After virtualization of hardware system we can install different operating system on it and run different applications on those OS.

Usage: Hardware virtualization is mainly done for the server platforms, because controlling virtual machines is much easier than controlling a physical server.

2) Operating System Virtualization:

When the virtual machine software or virtual machine manager *(VMM) is installed on the Host operating system* instead of directly on the hardware system is known as operating system virtualization.

Usage: Operating System Virtualization is mainly used for testing the applications on different platforms of OS.

3) Server Virtualization:

When the virtual machine software or virtual machine manager *(VMM) is directly installed on the Server system* is known as server virtualization.

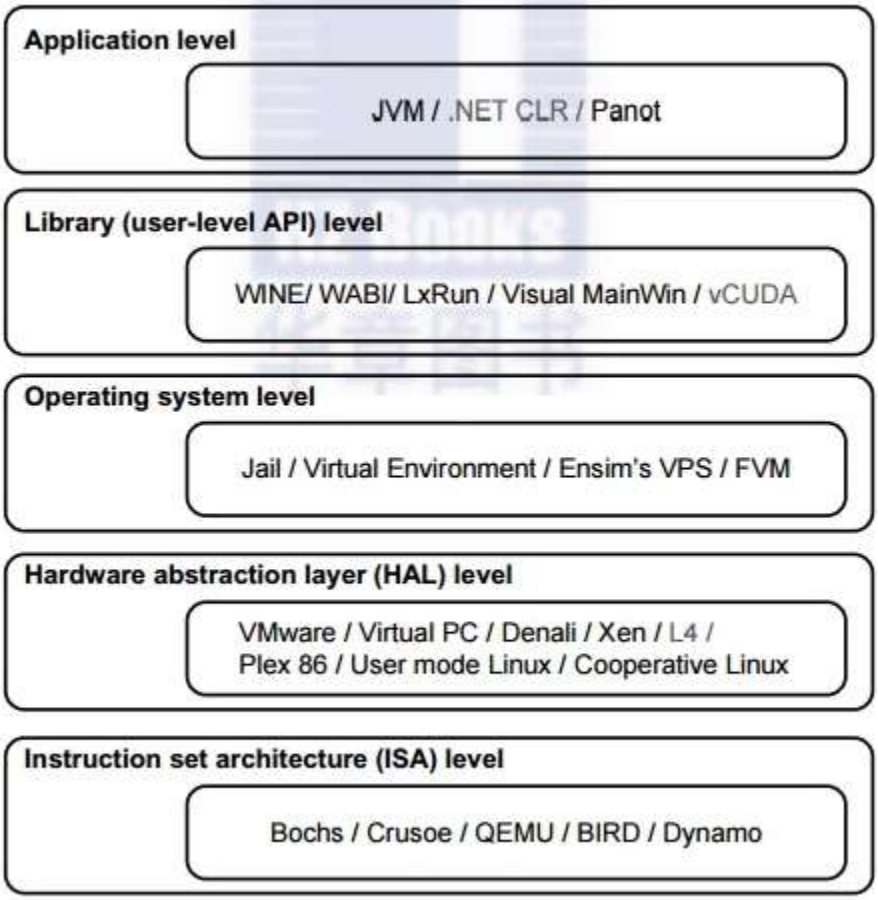
Usage: Server virtualization is done because a single physical server can be divided into multiple servers on the demand basis and for balancing the load.

4) Storage Virtualization:

Storage virtualization is the *process of grouping the physical storage from multiple network storage devices so that it looks like a single storage device*. Storage virtualization is also implemented by using software applications.

Usage: Storage virtualization is mainly done for back-up and recovery purposes.

Levels of Virtualization:



1. Instruction Set Architecture Level

At the ISA level, virtualization is performed by emulating a given ISA by the ISA of the host machine. For example, MIPS binary code can run on an x86-based host machine with the help of ISA emulation. With this approach, it is possible to run a large amount of legacy binary code writ-ten for various processors on any given new hardware host machine. Instruction set emulation leads to virtual ISAs created on any hardware machine.

2. Hardware Abstraction Level

Hardware-level virtualization is performed right on top of the bare hardware. On the one hand, this approach generates a virtual hardware environment for a VM. On the other hand, the process manages the underlying hardware through virtualization. The idea is to virtualize a computer's resources, such as its processors, memory, and I/O devices. The intention is to upgrade the hardware utilization rate by multiple users concurrently. The idea was implemented in the IBM VM/370 in the 1960s. More recently, the Xen hypervisor has been applied to virtualize x86-based machines to run Linux or other guest OS applications.

3. Operating System Level

This refers to an abstraction layer between traditional OS and user applications. OS-level virtualization creates isolated containers on a single physical server and the OS instances to utilize the hard-ware and software in data centers. The containers behave like real servers. OS-level virtualization is commonly used in creating virtual hosting environments to allocate hardware resources among a large number of mutually distrusting users.

4. Library Support Level

Most applications use APIs exported by user-level libraries rather than using lengthy system calls by the OS. Since most systems provide well-documented APIs, such an interface becomes another candidate for virtualization. Virtualization with library interfaces is possible by controlling the communication link between applications and the rest of a system through API hooks. The software tool WINE has implemented this approach to support Windows applications on top of UNIX hosts. Another example is the vCUDA which allows applications executing within VMs to leverage GPU hardware acceleration.

5. User-Application Level

Virtualization at the application level virtualizes an application as a VM. On a traditional OS, an application often runs as a process. Therefore, application-level virtualization is also known as process-level virtualization. The most popular approach is to deploy high level language

(HLL).VMs. In this scenario, the virtualization layer sits as an application program on top of the operating system, and the layer exports an abstraction of a VM that can run programs written and compiled to a particular abstract machine definition. Any program written in the HLL and compiled for this VM will be able to run on it. The Microsoft .NET CLR and Java Virtual Machine (JVM) are two good examples of this class of VM.

Advantages of Virtualization:

1. Resource optimization
2. Save resource and money
3. Enhance security
4. Easy disaster recovery

Conclusion:

Title: Study and implementation of infrastructure as Service using Open Stack.

OpenStack:

OpenStack is a free and open source, cloud computing software platform that is widely used in the deployment of infrastructure-as-a-Service (IaaS) solutions. The core technology with OpenStack comprises a set of interrelated projects that control the overall layers of processing, storage and networking resources through a data center that is managed by the users using a Web-based dashboard, command-line tools, or by using the Restful API. Currently, OpenStack is maintained by the OpenStack Foundation, which is a non-profit corporate organization established in September 2012 to promote OpenStack software as well as its community. Many corporate giants have joined the project, including GoDaddy, Hewlett Packard, IBM, Intel, Mellanox, Mirantis, NEC, NetApp, Nexenta, Oracle, Red Hat, SUSE Linux, VMware, Arista Networks, AT&T, AMD, Avaya, Canonical, Cisco, Dell, EMC, Ericsson, Yahoo!, etc.

OpenStack users

• AT&T	• Purdue University
• Stockholm University	• Red Hat
• SUSE	• CERN
• Deutsche Telekom	• HP Converged Cloud
• HP Public Cloud	• Intel
• KT (formerly Korea Telecom)	• NASA
• NSA	• PayPal
• Disney	• Sony
• Rackspace Cloud	• SUSE Cloud Solution
• Wikimedia Labs	• Yahoo!
• Walmart	• Opera Software

OpenStack releases with the components included
OpenStack Austin - Nova, Swift
OpenStack Bexar - Nova, Glance, Swift
OpenStack Cactus - Nova, Glance, Swift
OpenStack Diablo - Nova, Glance, Swift
OpenStack Essex - Nova, Glance, Swift, Horizon, Keystone
OpenStack Folsom - Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder
OpenStack Grizzly - Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder
OpenStack Havana - Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Heat, Ceilometer
OpenStack Icehouse - Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Heat, Ceilometer, Trove

OpenStack Computing Components:

OpenStack has a modular architecture that controls large pools of compute, storage and networking resources.

Compute (Nova):

OpenStack Compute (Nova) is the fabric controller, a major component of Infrastructure as a Service (IaaS), and has been developed to manage and automate pools of computer resources. It works in association with a range of virtualization technologies. It is written in Python and uses many external libraries such as Eventlet, Kombu and SQLAlchemy.

Object storage (Swift):

It is a scalable redundant storage system, using which objects and files are placed on multiple disks throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster. OpenStack Swift replicates the content from other active nodes to new locations in the cluster in case of server or disk failure.

Block storage (Cinder):

OpenStack block storage (Cinder) is used to incorporate continual block-level storage devices for usage with OpenStack compute instances. The block storage system of OpenStack is used to manage the creation, mounting and un mounting of the block devices to servers. Block storage is integrated for performance-aware scenarios including database storage, expandable file systems or providing a server with access to raw block level storage. Snapshot management in OpenStack provides the authoritative functions and modules for the back-up of data on block

storage volumes. The snapshots can be restored and used again to create a new block storage volume.

Networking (Neutron):

Formerly known as Quantum, Neutron is a specialised component of OpenStack for managing networks as well as network IP addresses. OpenStack networking makes sure that the network does not face bottlenecks or any complexity issues in cloud deployment. It provides the users continuous self-service capabilities in the network' s infrastructure. The floating IP addresses allow traffic to be dynamically routed again to any resources in the IT infrastructure, and therefore the users can redirect traffic during maintenance or in case of any failure. Cloud users can create their own networks and control traffic along with the connection of servers and devices to one or more networks. With this component, OpenStack delivers the extension framework that can be implemented for managing additional network services including intrusion detection systems (IDS), load balancing, firewalls, virtual private networks (VPN) and many others.

Figure 1: OpenStack

Dashboard (Horizon):

The OpenStack dashboard (Horizon) provides the GUI (Graphical User Interface) for the access, provision and automation of cloud-based resources. It embeds various third party products and services including advance monitoring, billing and various management tools.

Identity services (Keystone):

Keystone provides a central directory of the users, which is mapped to the OpenStack services they are allowed to access. It refers and acts as the centralized authentication system across the cloud operating system and can be integrated with directory services like LDAP. Keystone supports various authentication types including classical username and password credentials, token-based systems and other log-in management systems.

Image services (Glance):

OpenStack Image Service (Glance) integrates the registration, discovery and delivery services for disk and server images. These stored images can be used as templates. It can also be used to store and catalogue an unlimited number of backups. Glance can store disk and server images in different types and varieties of back-ends, including Object Storage.

Telemetry (Ceilometer):

OpenStack telemetry services (Ceilometer) include a single point of contact for the billing systems. These provide all the counters needed to integrate customer billing across all current and future OpenStack components.

Orchestration (Heat):

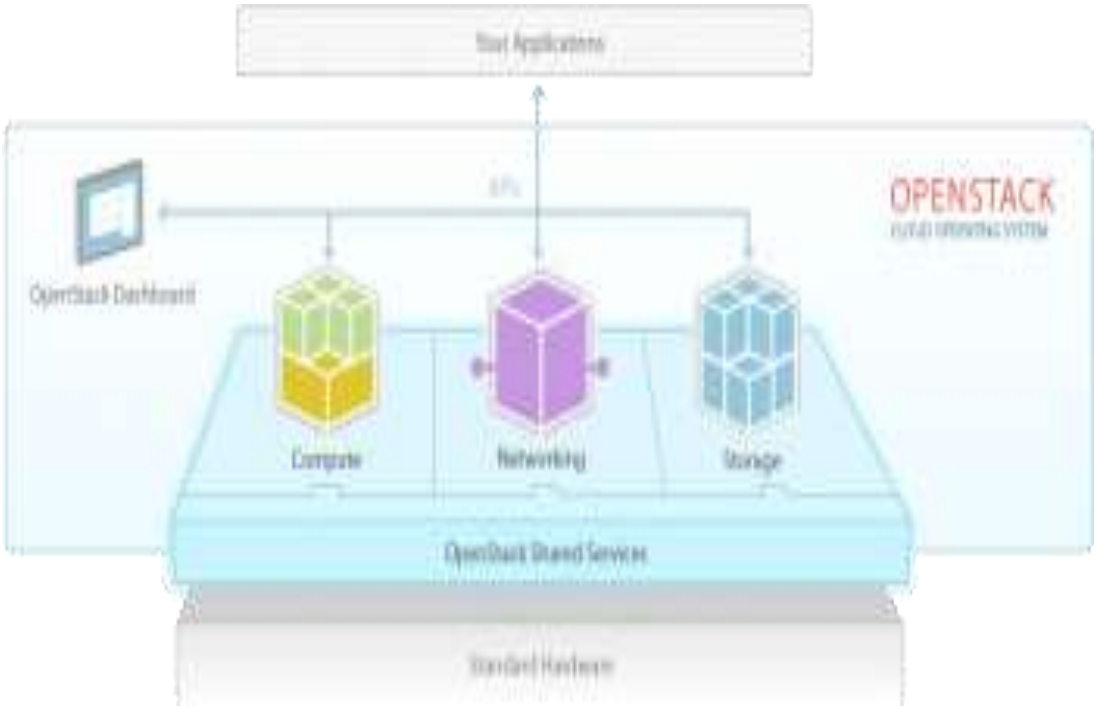
Heat organizes a number of cloud applications using templates with the help of the OpenStack-native REST API and a Cloud Formation-compatible Query API.

Database (Trove):

Trove is used as database-as-a-service (DaaS), which integrates and provisions relational and non-relational database engines.

Elastic Map Reduce (Sahara):

Sahara is the specialized service that enables data processing on OpenStack-managed resources, including the processing with Apache Hadoop.



Deployment of OpenStack using DevStack:

DevStack is used to quickly create an OpenStack development environment. It is also used to demonstrate the starting and running of OpenStack services, and provide examples of using them from the command line. DevStack has evolved to support a large number of configuration options and alternative platforms and support services. It can be considered as the set of scripts which install all the essential OpenStack services in the computer without any additional software or configuration. To implement DevStack, first download all the essential packages, pull in the OpenStack code from various OpenStack projects, and set everything for the deployment.

To install OpenStack using DevStack, any Linux-based distribution with 2GB RAM can be used to start the implementation of IaaS.

Here are the steps that need to be followed for the installation.

1. Install Git

```
$ sudo apt-get install git
```

2. Clone the DevStack repository and change the directory. The code will set up the cloud infrastructure.

```
$ git clone http://github.com/openstack-dev/devstack
$ cd devstack/

/devstack$ ls

accrc exercises HACKING.rst rejoin-stack.sh tests
AUTHORS exercise.sh lib run_tests.sh tools
clean.sh extras.d LICENSE samples unstack.sh
driver_certs files localrc stackrc
eucarc functions openrc stack-screenrc
exerciserc functions-common README.md stack.sh
```

stack.sh, *unstack.sh* and *rejoin-stack.sh* are the most important files. *stack.sh* script is used to set up DevStack.unstack.sh is used to destroy the DevStack setup.
If you are on the earlier execution of *./stack.sh*, the environment can be brought up by executing the rejoin_*stack.sh* script.

3. Execute the stack.sh script:

/devstack$ ./stack.sh
Here, the MySQL database password is entered. There’ s no need to worry about the installation of MySQL separately on this system. We have to specify a password and this script will install MySQL, and use this password there.
Finally, we will have the script ending as follows:

```
+ merge_config_group /home/r/devstack/local.conf post-extra
+ local localfile=/home/r/devstack/local.conf
+ shift
+ local matchgroups=post-extra
+ [[ -r /home/r/devstack/local.conf ]]
+ return 0
+ [[ -x /home/r/devstack/local.sh ]]
+ service_check
+ local service
+ local failures
+ SCREEN_NAME=stack
+ SERVICE_DIR=/opt/stack/status
+ [[ ! -d /opt/stack/status/stack ]]
++ ls ‘  /opt/stack/status/stack/*.failure’
++ /bin/true
+ failures=
+ ‘  [‘   -n ‘  ’   ‘  ]’
+ set +o xtrace
```

- Horizon is now available at *http://1.1.1.1/*
- Keystone is serving at *http://1.1.1.1:5000/v2.0/*
- Examples on using the *novaclient* command line are in *exercise.sh*
- The default users are: admin and demo
- The password: nova
- This is your host IP: *1.1.1.1*

After all these steps, the machine becomes the cloud service providing platform. Here, 1.1.1.1 is the IP of my first network interface.

We can type the host IP provided by the script into a browser, in order to access the dashboard ‘ Horizon’. We can log in with the username ‘ admin’ or ‘ demo’ and the password ‘ admin’.
You can view all the process logs inside the screen, by typing the following command:

```
$ screen -x
```

Executing the following will kill all the services, but it should be noted that it will not delete any of the code.
To bring down all the services manually, type:

```
$ sudo killall screen
```

localrc configurations

localrc is the file in which all the local configurations (local machine parameters) are maintained. After the first successful *stack.sh* run, you will see that a *localrc* file gets created with the configuration values you specified while running that script.

The following fields are specified in the *localrc* file:
DATABASE_PASSWORD
RABBIT_PASSWORD
SERVICE_TOKEN
SERVICE_PASSWORD
ADMIN_PASSWORD

If we specify the option *OFFLINE=True* in the *localrc* file inside DevStack directory, and if after specifying this, we run *stack.sh*, it will not check any parameter over the Internet. It will set up DevStack using all the packages and code residing in the local system. In the phase of code development, there is need to commit the local changes in the */opt/stack/nova* repository

before restack (re-running stack.sh) with the *RECLONE=yes option.* Otherwise, the changes will not be committed.

To use more than one interface, there is a need to specify which one to use for the external IP using this configuration:

HOST_IP=xxx.xxx.xxx.xxx

Cinder on DevStack

Cinder is a block storage service for OpenStack that is designed to allow the use of a reference implementation (LVM) to present storage resources to end users that can be consumed by the OpenStack Compute Project (Nova). Cinder is used to virtualise the pools of block storage devices. It delivers end users with a self-service API to request and use the resources, without requiring any specific complex knowledge of the location and configuration of the storage where it is actually deployed.

All the Cinder operations can be performed via any of the following:

1. CLI (Cinder’s *python-cinderclient* command line module)
2. GUI (Using OpenStack’s GUI project *horizon*)
3. Direct calling of Cinder APIs

Creation and deletion of volumes: To create a 1 GB Cinder volume with no name, run the following command:

```
$ cinder create 1
```

To see more information about the command, just type cinder help *<command>*

```
$ cinder help create

usage: cinder create [--snapshot-id <snapshot-id>]
[--source-volid <source-volid>] [--image-id <image-id>]
[--display-name <display-name>]
[--display-description <display-description>]
[--volume-type <volume-type>]
[--availability-zone <availability-zone>]
[--metadata [<key=value> [<key=value> ...]]]
<size>
Add a new volume.
Positional arguments:
<size> Size of volume in GB
Optional arguments:
--snapshot-id <snapshot-id>
```

Create volume from snapshot id (Optional,
Default=None)
--source-volid <source-volid>
Create volume from volume id (Optional, Default=None)
--image-id <image-id>
Create volume from image id (Optional, Default=None)
--display-name <display-name>
Volume name (Optional, Default=None)
--display-description <display-description>
Volume description (Optional, Default=None)
--volume-type <volume-type>
Volume type (Optional, Default=None)
--availability-zone <availability-zone>
Availability zone for volume (Optional, Default=None)
--metadata [<key=value> [<key=value> ...]]
Metadata key=value pairs (Optional, Default=None)

To create a Cinder volume of size 1GB with a name, using *cinder create –display-name myvolume:*

```
$ cinder create --display-name myvolume 1
+------------------+--------------------------------------+
| Property | Value |
+------------------+------------------------------------- -+
| attachments | [] |
| availability_zone | nova |
| bootable | false |
| created_at | time | | display_description | None |
| display_name | myvolume | | id | id |
| metadata | {} |
| size | 1 |
| snapshot_id | None |
| source_volid | None |
| status | creating |
| volume_type | None |
+----------------------+----------------------------------+
```

To list all the Cinder volumes, using *cinder list:*

```
$ cinder list
ID Status Display Name Size Volume type Bootable Attached To
id1 Available Myvolume 1 None False
id2 Available None 1 None False
```

To delete the first volume (the one without a name), use the *cinder delete <volume_id>*command. If we execute *cinder* list really quickly, the status of the volume going to ‘ deleting’ can be seen, and after some time, the volume will be deleted:

```
$ cinder delete id2

$ cinder list
ID Status Display Name Size Volume type Bootable Attached To
id1 Available Myvolume 1 None False
id2 Deleting None 1 None False
```

Volume snapshots can be created as follows:

```
$ cinder snapshot-create id2
+---------------------+--------------------------------+
| Property | Value |
+---------------------+-----------------------------------+
| created_at | TimeStamp |
| display_description | None |
| display_name | None |
| id | snapshot2 |
| metadata | {} |
| size | 1 |
| status | creating |
| volume_id | id2 |
+---------------------+--------------------------------+
```

All the snapshots can be listed as follows:

```
$ cinder snapshot-list

ID Volume ID Status Display Name Size
Snapshotid1 id2 Available None 1
```

You can also create a new volume of 1GB from the snapshot, as follows:

```
$ cinder create --snapshot-id snapshotid1 1
+---------------------+--------------------------------------
| Property | Value
+---------------------+--------------------------------------
| attachments | [] |
| availability_zone | nova |
| bootable | false |
| created_at | creationtime |
| display_description | None |
| display_name | None |
| id | v1 |
| metadata | {} |
| size | 1 |
| snapshot_id | snapshotid1 |
| source_volid | None |
| status | creating |
| volume_type | None |
+---------------------+--------------------------------------+
```

There are lots of functions and features available with OpenStack related to cloud deployment. Depending upon the type of implementation, including load balancing, energy optimization, security and others, the cloud computing framework OpenStack can be explored a lot.

Title:

Write a program for Web feed using PHP and HTML.

Theory:

Web Application and Cloud Computing

The services are accessible anywhere in the world, with the cloud appearing as a single point of access for all the computing needs of consumers. New advances in processors, virtualization technology, disk storage, broadband internet access and fast, inexpensive servers have all combined to make cloud computing a compelling paradigm. Cloud computing allows users and companies to pay for and use the software and storage that they need, when they need then and as wireless broadband connection options grow, where they need them. This type of software deployment is called Software as a Service (SaaS).

Many of the underlying technologies such as grid computing, peer-to-peer computing have a direct contribution to cloud computing. In order to understand what type of components exist in a cloud, we first need to enumerate the typical components of an application development. The components excluding human resources are i) infrastructure resources ii) software resources, iii) application resources and iv) business processes.

In the cloud computing paradigm, all of the above components are treated as services and are in the “cloud”; users do not have to invest or pay huge licensing fees to won any of the above resources. Infrastructure resources are storage, computing power and so forth, which can take advantage of already existing technologies such as grid computing. The software resources include application servers, database servers, IDE and so on. The application resources include applications deployed as SaaS for example Google docs. The business process resources can be standard set of common business utilities given as services to clients. Example is ERP software such as SAP and Oracle providing standard business workflows in the cloud. Some of the major players in cloud computing are Amazon, Google, IBM, Joint, Microsoft and Sales Force. Current cloud computing services are storage services, spam filtering, performing applications in high level programming languages such as Java or the use of some kind of database. In 2008, Google has released Google App Engine, a cloud-based platform used for running applications both

individuals and businesses. Microsoft has released Windows Azure, a cloud-based operating system, for the Community technology Preview.

What is Web feed?

A web feed (or news feed) is a data format used for providing users with frequently updated content. Content distributors syndicate a web feed, thereby allowing users to subscribe to it. Making a collection of web feeds accessible in one spot is known as aggregation, which is performed by an aggregator. A web feed is also sometimes referred to as a syndicated feed.

A typical scenario of web feed use is: a content provider publishes a feed link on their site which end users can register with an aggregator program (also called a feed reader or a news reader) running on their own machines; doing this is usually as simple as dragging the link from the web browser to the aggregator. When instructed, the aggregator asks all the servers in its feed list if they have new content; if so, the aggregator either makes a note of the new content or downloads it. Aggregators can be scheduled to check for new content periodically. Web feeds are an example of pull technology, although they may appear to push content to the user.

The kinds of content delivered by a web feed are typically HTML (webpage content) or links to webpages and other kinds of digital media. Often when websites provide web feeds to notify users of content updates, they only include summaries in the web feed rather than the full content itself. Web feeds are operated by many news websites, weblogs, schools, and podcasters.

Benefits

1. Users do not disclose their email address when subscribing to a feed and so are not increasing their exposure to threats associated with email: spam, viruses, phishing, and identity theft.
2. Users do not have to send an unsubscribe request to stop receiving news. They simply remove the feed from their aggregator.

3. The feed items are automatically sorted in that each feed URL has its own sets of entries (unlike an email box where messages must be sorted by user-defined rules and pattern matching).

Conclusion:

Title:

Write a Program to Create, Manage and groups User accounts in own Cloud by Installing Administrative Features.

Theory:

What is OwnCloud?

OwnCloud is a suite of client–server software for creating and using file hosting services. OwnCloud is functionally very similar to the widely used Drop box, with the primary functional difference being that the Server Edition of ownCloud is free and open-source, and thereby allowing anyone to install and operate it without charge on a private server. It also supports extensions that allow it to work like Google Drive, with online document editing, calendar and contact synchronization, and more. Its openness avoids enforced quotas on storage space or the number of connected clients, instead having hard limits (like on storage space or number of users) defined only by the physical capabilities of the server.

The development of ownCloud was announced in January 2010, in order to provide a free software replacement to proprietary storage service providers. The company was founded in 2011 and forked the code away from KDE to github.

Overview:

Design:

For desktop machines to synchronize files with their ownCloud server, desktop clients are available for PCs running Windows, macOS, FreeBSD or Linux. Mobile clients exist for iOS and Android devices.

Files and other data (such as calendars, contacts or bookmarks) can also be accessed, managed, and uploaded using a web browser without any additional software.

Any updates to the file system are pushed to all computers and mobile devices connected to a user's account.Encryption of files may be enforced by the server administrator.

The ownCloud server is written in the PHP and JavaScript scripting languages. For remote access, it employs sabre/dav, an open-source WebDAV server.

OwnCloud is designed to work with several database management systems, including SQLite, MariaDB, MySQL, Oracle Database, and PostgreSQL.

Features:-

ownCloud files are stored in conventional directory structures, and can be accessed via WebDAV if necessary. User files are encrypted both at rest and during transit. ownCloud can synchronise with local clients running Windows (Windows XP, Vista, 7 and 8), macOS (10.6 or later), or various Linux distributions.

ownCloud users can manage calendars (CalDAV), contacts (CardDAV) scheduled tasks and streaming media (Ampache) from within the platform.

From the administration perspective, ownCloud permits user and group administration (via OpenID or LDAP). Content can be shared by defining granular read/write permissions between users and/or groups. Alternatively, ownCloud users can create public URLs when sharing files. Logging of file-related actions is available in the Enterprise and Education service offerings.

Furthermore, users can interact with the browser-based ODF-format word processor, bookmarking service, URL shortening suite, gallery, RSS feed reader and document viewer tools from within ownCloud. For additional extensibility, ownCloud can be augmented with "one-click" applications and connection to Dropbox, Google Drive and Amazon S3.

All ownCloud clients (Desktop, iOS, Android) support the OAuth 2 standard for Client Authentication.

Enterprise Features:-

For Enterprise customers, ownCloud GmbH offers apps with additional functionality. They are mainly useful for large organizations with more than 500 users. An Enterprise subscription includes support services.

Commercial features include End-to-end encryption, Ransomware and Antivirus protection, Branding, Document Classification, Single-Sign-On via Shibboleth/SAML, and a lot more.[16]

Distribution:-

ownCloud server and clients may be downloaded from the ownCloud website and from third-party repositories, such as Google Play and Apple iTunes, and repositories maintained by Linux distributions.

In 2014, a dispute arose between ownCloud and Ubuntu regarding the latter allegedly neglecting maintenance of packages, resulting in the temporary removal of ownCloud from the Ubuntu repository.

ownCloud has been integrated with the GNOME desktop. Additional projects that use or link to ownCloud include a Raspberry Pi project to create a cloud storage system using the Raspberry Pi's small, low-energy form-factor.

Conclusion:

Title:

Case study on Amazon EC2 to learn about Amazon EC2. Amazon EC2 Elastic Compute cloud is a central part of Amazon.com's cloud computing platform, Amazon Web Services. How EC2 allows users torrent virtual computers on which to learn run their own computer applications.

Case Study: Amazon EC2.

Abstract:

- Amazon Elastic compute cloud (EC2) forms a central part of Amazon.com's cloud computing platform, Amazon Web Services (AWS), by following users to rent virtual computers on which to run their own computer applications.
- EC2 encourages scalable development of applications by providing a web service through which a user can robot boot an Amazon Machine Image (AMI) to configure a virtual machine, which Amazon calls an "Instance", containing any software desired.
- A user can create, launch and terminate server-instances as needed, paying by the second for active servers - hence the term "elastic" EC2 provides users with control over the geographical location of instances that allows for latency optimization and high levels of redundancy.
- In November 2010, Amazon switches as its own retail website to use EC2 and AWS.

KEYWORDS:

- Introduction

- Architecture

Introduction:

EC2 uses Simple Storage Services (S3) for Storage of data. Users can hire suitable amount CPU power, Storage and memory without any upfront commitment. Users can control the entire Software Stack from kernel upwards.

Architecture:

The architecture has two components.

- S3
- EC2

S3 and EC2 is used for Storage and computing purposes.

Simple Storage Service:

- S3 can be through as a globally available distributed hash value table with high-level access control. Data is stored in pairs of name and value.
- Names are like UNIX file names and the value can be object having size up-to 5GB with up-to 4K of metadata for each object. Into the global namespace must fit all objects in Amazon's S3. This namespace contain a "bucketname" and an "object name".
- An AWS (Amazon Web services) account can have maximum of 100 buckets. Bucket names are like user names in traditional email account and provided by Amazon on first come first serve basis. This namespace Data to S3

can be sent by SOAP based API or with raw HTTP "PUT" commands.

- Data can be recovered using SOAP HTTP or BitTorrent. While using BitTorrent the S3 system operates as both tracker and the initial seeder. There are also some tools available which enables the users to view S3 as a remove file system.
- Upload download rate from and to S3 is not that much exiting one developer from Germany reported experiencing 10-100kBps. This rate can go upto 1-2MBps on the higher side depending on the time of the day.
- Although the speed is not that much fascinating it is good enough for delivering web objects and for backup purposes although for doing computation it is not suitable. Amazon S3 has a very impressive support for privacy, integrity and short term availability.
- Long term availability is unknown as this depends on the internal commitment of Amazon data centers. Data privacy can be obtained by encrypting the data to be stored. But this encryption is to be done by the user before storing the data in S3.
- One can use SSL with HTTPS to connect to S3 for more security but this uses of SSL increases upload/download time also. Data integrity can be achieved by checking end to end MD5 checking. When an object is stored into S3 then it returns MD5 of that object.
- One can easily check it with previously computed hash value to guarantee data integrity. Short term availability depends upon the Amazon's connectivity and load on its server at that

instant.

- Once the data is actually in the S3 then it is Amazon's responsibility to take care of it's availability. They claim that the data is backup on multiple hard drives in multiple data centers but doesn't guarantee this by any service Level Agreement.
- There is no backup or recovery mechanism if the user accidentally deletes any data. Amazon has a very impressive scheme of authentication in comparison to other cloud services. Every AWS account has an Access key ID and a Secret key.
- The ID is of 20 characters and the key is a 41 character string. When signing HMAC is first computed for the sign request parameters using that key. And in the Amazon server that HMAC is again computed a compared with the value previously computed in client side. These requests also include timestamp prevent reply attacks.

Elastic Compute Cloud:

- As the name implies EC2 are cloud of computers to the users with flexibility of choosing the configuration of the virtual machine like RAM size, local disk size, propcessor speeds etc.
- Machines that deliver EC2 services are actually virtual machines running on top of XEN platform. Users can machine store a disk image inside S3 and create a virtual machine in EC2 using tools provided by Amazon This virtual machine can be easily instantiated using a java program and can also be monitored. As EC2.

is based on XEN it supports by any linux distribution as well as other OSs. Amazon does not promise about reliability of the EC2 Computers.

- Any machine can crash at any moment and they are not backed up. Although these machine generally don't crash according to the experience of the users but it is safe to use S3 to store information which is more reliable and replicated service. EC2 security model is similar to that of S3. The only difference is that the commands are signed with an X 509 private key. But this key is downloaded from AWS account so the security depends fundamentally on the AWS username and password.

Title:

Case study on microsoft Azure to learn about microsoft Azure is a cloud computing platform and infrastructure, created by microsoft, for building, deploying and managing applications and services through a global network of microsoft managed datacenters. How it work, different services provided by it.

Case study: Microsoft azure.

Abstract:

- Microsoft Azure is a cloud computing service created by Microsoft for building, testing, deploying and managing applications and services through a global network of Microsoft managed data centers. It provides Software as a Service (SaaS) platform as a Service (PaaS) and information as a Service (IaaS) and support many different programming languages, tools and frameworks, including both Microsoft specific and third-party software and systems.
- Azure was announced in October 2008, started with codename "Project Red Dog", and released in February 1, 2010 as "Windows Azure" before being renamed "Microsoft Azure" on March 25, 2014.

KEYWORDS:

- Introduction
- Architecture
- Services

Introduction:

- Microsoft Azure uses a specialized operating system, called Microsoft Azure, to run its its "febric layer": a cluster hosted at microsoft's data centers that manages computing and storage resources of computers and provisions the resources (or a subset of them) to applications running on top of microsoft Azure.
- Microsoft Azure has been described as a "cloud layer" on top of a number of Windows Server systems, Which use Windows Server 2008 and a customized version of Hyper-V known as the microsoft Azure Hypervisor to provide virtualization of services.
- Scaling and reliability are controlled by the microsoft Azure Fabic controller, Which ensures the services and environment do not fail if one or more of the servers fails within the microsoft data center, and which also provides the management of the user's Web application Such as memory allocation and load balancing.
- Azure provides an API built on REST, HTTP, and XML that allows a developer to interact with the services provided by microsoft Azure, microsoft also provides a client-side managed class library that encapsulates the function of interacting with the services with microsoft Visual Studio, Git and Eclipse.

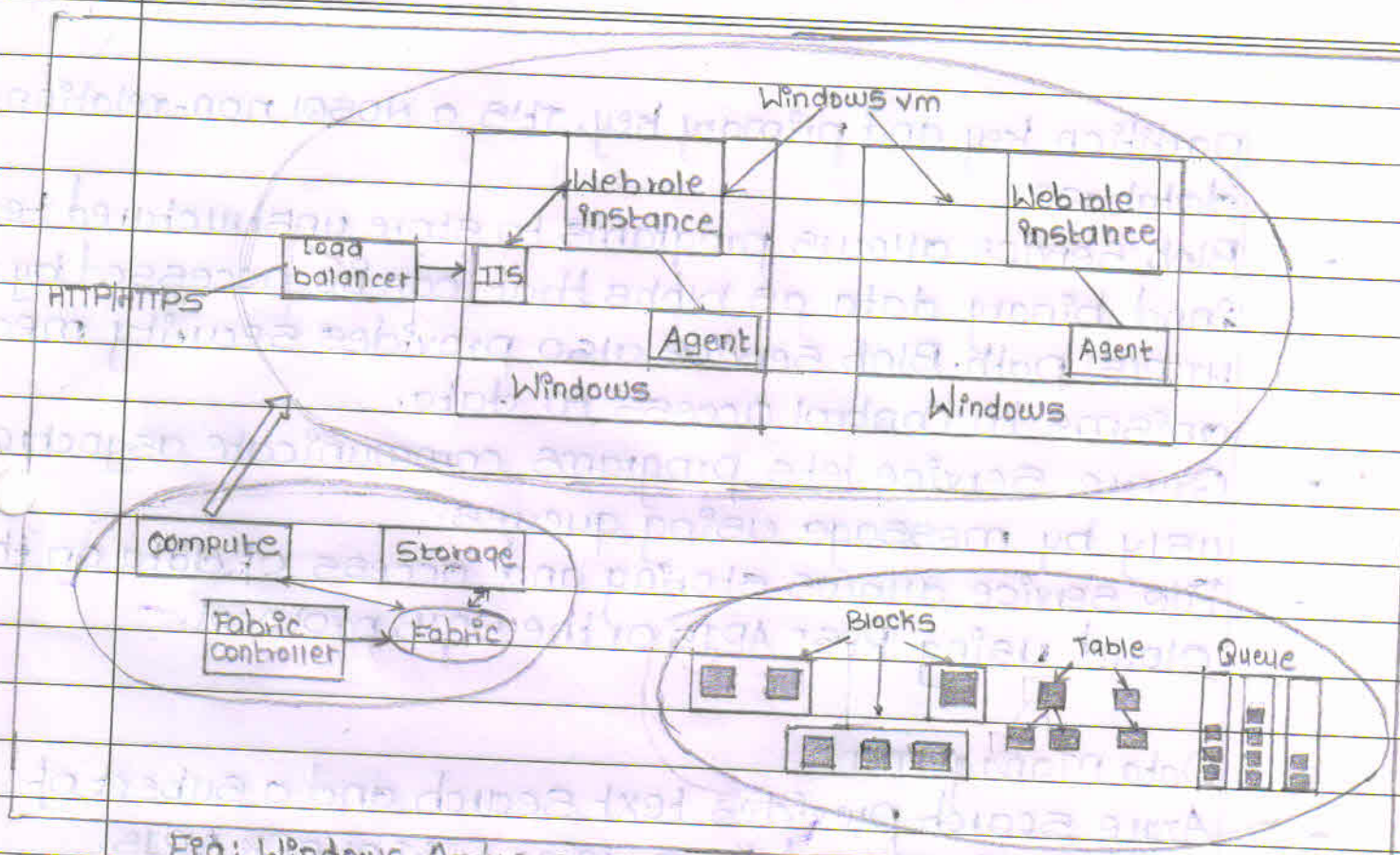


Fig: Windows Azure component Architecture.

Services:

Mobile Services:

- Mobile engagement collects real-time analytics that highlight users' behavior. It also provides push notifications to mobile devices.
- HockeyApp can be used to develop, distribute and beta-test mobile apps.

Storage Services:

- Storage services provides REST and SDK APIs for storing and accessing data on the cloud.
- Table service lets programs store structured text in partitioned collections of entities that are accessed by

partition key and primary key. It's a NOSQL non-relational database.

- Blob Service allows programs to store unstructured text and binary data as blobs that can be accessed by a HTTP(S) path. Blob service also provides security mechanisms to control access to data.
- Queue Service lets programs communicate asynchronously by message using queues.
- File service allows storing and access of data on the cloud using REST APIs or the SMB protocol.

Data Management:

- Azure search provides text search and a subset of OData's structured files using REST or SDK APIs.
- Cosmos DB is a NOSQL database service that implements a subset of file the SQL SELECT statement on JSON documents.
- Redis cache is a managed implementation of Redis.
- StorSimple manages storage task between on-premises devices and cloud storage.
- SQL database, formerly known as SQL Azure database, works to create, scale and extended applications into the cloud using Microsoft SQL server technology. It is also Integrates with Active Directory and Microsoft System Center and Hadoop.
- SQL Data Warehouse is a data warehousing service designed to handle computational and data intensive queries on datasets exceeding 1TB.
- Azure data Factory, is a data integration service that

allows creation of data-driven Workflows in the cloud for orchestrating and automating data movement and data transformation.

- Azure Data lake is scalable data storage and analytic service for big-data analytics Workloads that require developers to run massively parallel queries.
- Azure HDInsight is a big data relevant service that deploys Hortonworks Hadoop on microsoft Azure, and supports the creation of Hadoop clusters using linux with Ubuntu.
- Azure stream Analytics is a serverless scalable event processing engine that enables users to develop and run real-time analytics on multiple streams of data from sources such as devices, sensors, Web sites, social media, and other applications.

Messaging:

- The microsoft Azure Service Bus allows applications running on Azure premises or off premises devices to communicate with Azure. This helps to build people scalable and reliable applications in a service-oriented architecture (SOA). The Azure service bus support four different types of communication mechanisms:
- Even Hubs, Which provide event and telemetry ingress to the cloud at massive scale, with low latency and high reliability. For example an event hub can be used to track data from cell phones such as a GPS location coordinate in real time.
- Queues which allows one-dimentional communication. A sender application would send the message to the servi-

ce but queue, and a receiver would read from the service queue. Though there can be multiple readers for the queue only one would process a single message.

- Topics, which provides one-directional communication using a subscriber pattern. It is similar to a queue; however each subscriber will receive a copy of the message sent to a Topic. Optionally the subscriber can filter out messages based specific criteria defined by the subscriber.
- Relays, which provide bi-directional communication. Unlike queues and topics, a relay doesn't store in-flight messages in its own memory. Instead it just passes them on to the destination application.

Title:

Design and develop custom Application (Mini Project) using Salesforce Cloud.

Theory:

Introduction

Salesforce.com Inc. is an American cloud-based software company headquartered in San Francisco, California. Though the bulk of its revenue comes from a customer relationship management (CRM) product, Salesforce also sells a complementary suite of enterprise applications focused on customer service, marketing automation, analytics and application development.

Salesforce is the primary enterprise offering within the Salesforce platform. It provides companies with an interface for case management and task management, and a system for automatically routing and escalating important events. The Salesforce customer portal provides customers the ability to track their own cases, includes a social networking plug-in that enables the user to join the conversation about their company on social networking websites, provides analytical tools and other services including email alert, Google search, and access to customers' entitlement and contracts.

Lightning Platform

Lightning Platform (also known as Force.com) is a platform as a service (PaaS) that allows developers to create add-on applications that integrate into the main Salesforce.com application. These third-party applications are hosted on Salesforce.com's infrastructure.

Force.com applications are built using declarative tools, backed by Lightning and Apex (a proprietary Java-like programming language for Force.com) and Lightning and Visual force (a framework that includes an XML syntax typically used to generate HTML). The Force.com platform typically receives three complete releases a year. As the platform is provided as a service to its developers, every single development instance also receives all these updates.

Community Cloud

Community Cloud provides Salesforce customers the ability to create online web properties for external collaboration, customer service, channel sales, and other custom portals in their instance of Salesforce. Tightly integrated to Sales Cloud, Service Cloud, and App Cloud, Community Cloud can be quickly customized to provide a wide variety of web properties

Salesforce Sales Cloud

Salesforce Sales Cloud is a customer relationship management (CRM) platform designed to support sales, marketing and customer support in both business-to-business (B2B) and business-to-customer (B2C) contexts. Sales Cloud is a fully customizable product that brings all the customer information together in an integrated platform that incorporates marketing, lead generation, sales, customer service and business analytics and provides access to thousands of applications through the AppExchange. The platform is provided as Software as a Service (SaaS) for browser-based access; a mobile app is also available. A real-time social feed for collaboration allows users to share information or ask questions of the user community.Salesforce.com offers five versions of Sales Cloud on a per-user, per month basis, from lowest to highest: Group, Professional, Enterprise, Unlimited and Performance. The company offers three levels of support contracts: Standard Success Plan, Premier Success Plan and Premier+ Success Plan.

Create Custom Apps for Salesforce Classic

Create custom apps to give your Salesforce Classic users' access to everything they need all in one place.

If you're new to custom apps, we recommend using Lightning Platform quick start to create an app. With this tool, you can generate a basic working app in just one step.

If you've already created the objects, tabs, and fields you need for your app, follow these steps. With this option, you create an app label and logo, add items to the app, and assign the app to profiles.

1. From Setup, enter Apps in the Quick Find box, then select **Apps**.
2. Click **New**.
3. If the Salesforce console is available, select whether you want to define a custom app or a Salesforce console.
4. Give the app a name and description.

   An app name can have a maximum of 40 characters, including spaces.

5. Optionally, brand your app by giving it a custom logo.
6. Select which items to include in the app.
7. Optionally, set the default landing tab for your new app using the **Default Landing Tab** drop-down menu below the list of selected tabs. This determines the first tab a user sees when logging into this app.
8. Choose which profiles the app will be visible to.
9. Check the Default box to set the app as that profile's default app, meaning that new users with the profile see this app the first time they log in. Profiles with limits are excluded from this list.
10. Click **Save**

What is the difference between custom application and console application in sales force?

A custom application is a collection of tabs, objects etc that function together to solve a particular problem.

A console application uses a specific Salesforce UI - the console. Console applications are intended to enhance productivity by allowing everything to be done from a single, tabbed, screen**.**

Conclusion:

Title:

Assignment to install and configure Google App Engine.

Theory:

Introduction

Google App Engine is a web application hosting service. By “web application,” we mean an application or service accessed over the Web, usually with a web browser: storefronts with shopping carts, social networking sites, multiplayer games, mobile applications, survey applications, project management, collaboration, publishing, and all the other things we’re discovering are good uses for the Web. App Engine can serve traditional website content too, such as documents and images, but the environment is especially designed for real-time dynamic applications. Of course, a web browser is merely one kind of client: web application infrastructure is well suited to mobile applications, as well.

In particular, Google App Engine is designed to host applications with many simultaneous users. When an application can serve many simultaneous users without degrading performance, we say it scales. Applications written for App Engine scale automatically. As more people use the application, App Engine allocates more resources for the application and manages the use of those resources. The application itself does not need to know anything about the resources it is using.

The app engine is a Cloud-based platform, is quite comprehensive and combines infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS). The app engine supports the delivery, testing and development of software on demand in a Cloud computing environment that supports millions of users and is highly scalable.

The company extends its platform and infrastructure to the Cloud through its app engine. It presents the platform to those who want to develop SaaS solutions at competitive costs .Have you ever wondered as to who stands to benefit the most from the Google app engine? If you are a business SME or enterprise which owns any web-based application that needs to be scaled without any compromise on the performance then Google App Engine is a good fit. Companies like Best Buy and Khan Academy have chosen Google App Engine for their apps.

Google App Engine:

It is a platform-as-a-service (PaaS) Cloud computing platform that is fully managed and uses inbuilt services to run your apps. You can start development almost instantly after downloading the software development kit (SDK). You can go on to the developer's guide right away when you click on the language you wish to develop your app in.
As soon as you have signed up for a Cloud account, you can build your app:

- ✓ With the template/HTML package in Go
- ✓ With Jinja2 and webapp2 in Python
- ✓ With Cloud SQL in PHP
- ✓ With Maven in Java

Generally Available Features

These are covered by the depreciation policy and the service-level agreement of the app engine. Any changes made to such a feature are backward-compatible and implementation of such a feature is usually stable. These include data storage, retrieval, and search; communications; process management; computation; app configuration and management.

- ✓ Data storage, retrieval, and search include features such as HRD migration tool, Google Cloud SQL, logs, datastore, dedicated Memcache, blobstore, Memcache and search.
- ✓ Communications include features such as XMPP. channel, URL fetch, mail, and Google Cloud Endpoints.
- ✓ Process management includes features like scheduled tasks and task queue
- ✓ Computation includes images.
- ✓ App management and configuration cover app identity, users, capabilities, traffic splitting, modules, SSL for custom domains, modules, remote access, and multitenancy.

Advantages of Google App Engine:

- ✓ **Infrastructure for Security**

  Around the world, the Internet infrastructure that Google has is probably the most secure. There is rarely any type of unauthorized access till date as the application data and code are stored in highly secure servers.

  You can be sure that your app will be available to users worldwide at all times since Google has several hundred servers globally. Google's security and privacy policies are applicable to the apps developed using Google's infrastructure.

- ✓ **Scalability**

  For any app's success, this is among the deciding factors. Google creates its own apps using GFS, Big Table and other such technologies, which are available to you when you utilize the Google app engine to create apps. You only have to write the code for the app and Google looks after the testing on account of the automatic scaling feature that the app engine has. Regardless of the amount of data or number of users that your app stores, the app engine can meet your needs by scaling up or down as required.

- ✓ **Performance and Reliability**

  Google is among the leaders worldwide among global brands. So, when you discuss performance and reliability you have to keep that in mind. In the past 15 years, the company has created new benchmarks based on its services' and products' performance. The app engine provides the same reliability and performance as any other Google product.

- ✓ **Cost Savings**

  You don't have to hire engineers to manage your servers or to do that yourself. You can invest the money saved into other parts of your business.

- ✓ **Platform Independence**

  You can move all your data to another environment without any difficulty as there is not many dependencies on the app engine platform.

Conclusion:

Title:

Design an Assignment to retrieve, verify, and store user credentials using Firebase Authentication, the Google App Engine standard environment, and Google Cloud Data store.

Theory:

Firebase Authentication

Most apps need to know the identity of a user. Knowing a user's identity allows an app to securely save user data in the cloud and provide the same personalized experience across all of the user's devices.

Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more.

Firebase Authentication integrates tightly with other Firebase services, and it leverages industry standards like OAuth 2.0 and OpenID Connect, so it can be easily integrated with your custom backend.

How does it work?

To sign a user into your app, you first get authentication credentials from the user. These credentials can be the user's email address and password, or an OAuth token from a federated identity provider. Then, you pass these credentials to the Firebase Authentication SDK. Our backend services will then verify those credentials and return a response to the client.

After a successful sign in, you can access the user's basic profile information, and you can control the user's access to data stored in other Firebase products. You can also use the provided authentication token to verify the identity of users in your own backend services.

Authenticating Users on App Engine Using Firebase:

Now we show how to retrieve, verify, and store user credentials using Firebase Authentication, the Google App Engine standard environment, and Google Cloud Data store.

The document walks you through a simple note-taking application called Firenotes that stores users' notes in their own personal notebooks. Notebooks are stored per user, and identified by each user's unique Firebase Authentication ID. The application has the following components:

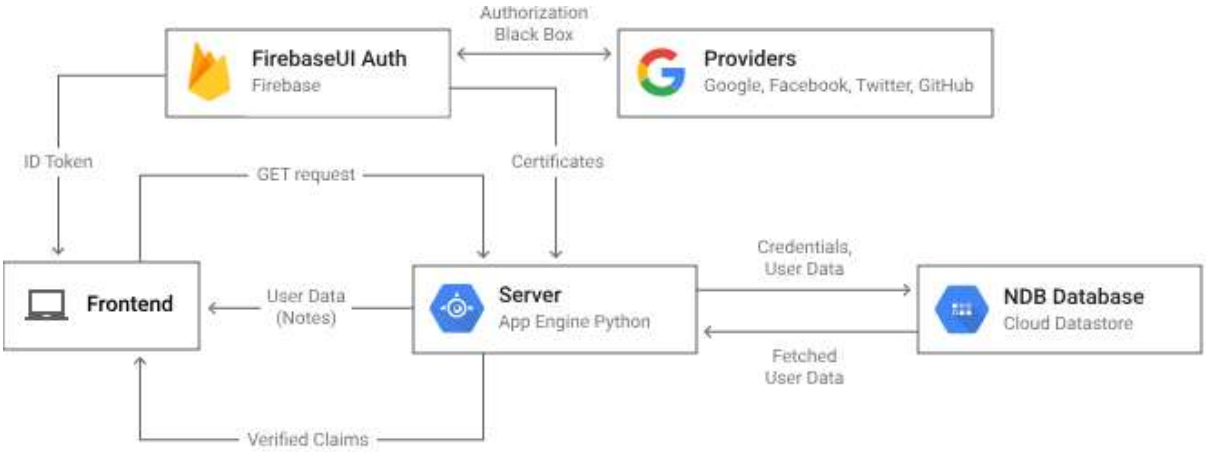
1. The frontend configures the sign-in user interface and retrieves the Firebase Authentication ID. It also handles authentication state changes and lets users see their notes.

2. FirebaseUI is an open-source, drop-in solution that handles user login, linking multiple providers to one account, recovering passwords, and more. It implements authentication best practices for a smooth and secure sign-in experience.

3. The backend verifies the user's authentication state and returns user profile information as well as the user's notes.

The application stores user credentials in Cloud Data store by using the NDB client library, but you can store the credentials in a database of your choice.

The following diagram shows how the frontend and backend communicate with each other and how user credentials travel from Firebase to the database.



Objectives:

- Configure the Firebase Authentication user interface.
- Obtain a Firebase ID token and verify it using server-side authentication.
- Store user credentials and associated data in Cloud Datastore.
- Query a database using the NDB client library.
- Deploy an app to App Engine.

Costs:

This tutorial uses billable components of Cloud Platform, including:

Google Cloud Data store

Conclusion:

Title:

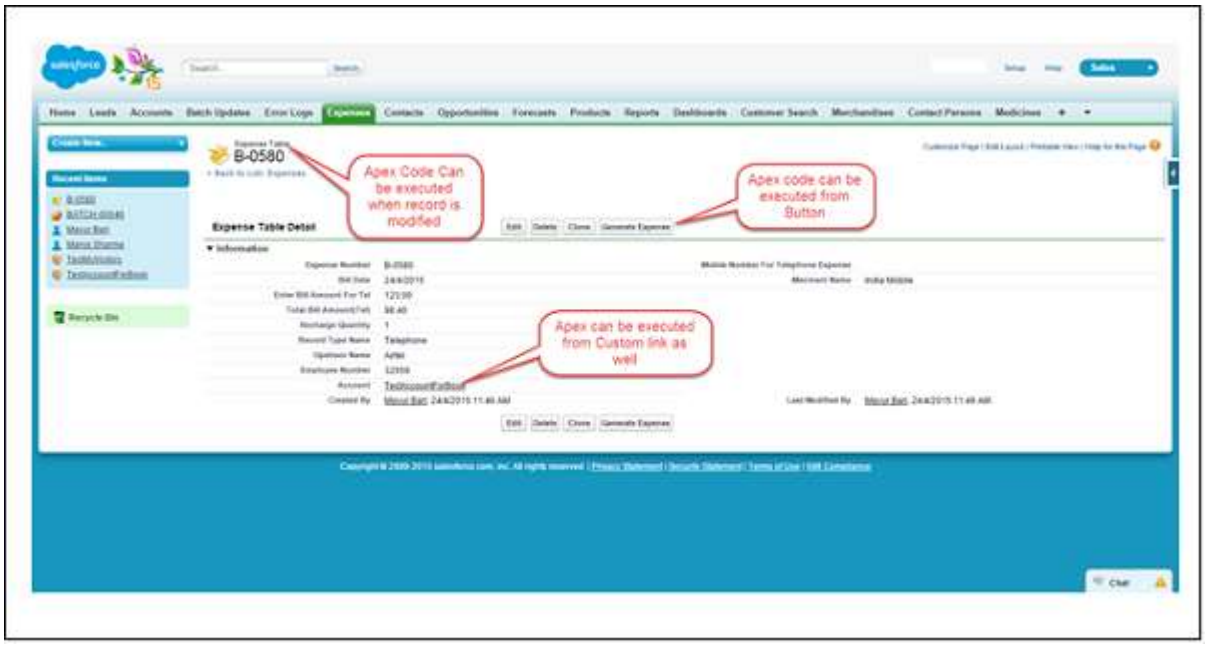
Creating an Application in SalesForce.com using Apex programming Language

Theory:

What is Apex?

Apex is a proprietary language developed by the Salesforce.com. As per the official definition, Apex is a strongly typed, object-oriented programming language that allows developers to execute the flow and transaction control statements on the Force.com platform server in conjunction with calls to the Force.com API.

It has a Java-like syntax and acts like database stored procedures. It enables the developers to add business logic to most system events, including button clicks, related record updates, and Visual force **pages. Apex** code can be initiated by Web service requests and from triggers on objects. Apex is included in Performance Edition, Unlimited Edition, Enterprise Edition, and Developer Edition.



Features of Apex as a Language

Let us now discuss the features of Apex as a Language −

- Integrated

  Apex has built in support for DML operations like INSERT, UPDATE, DELETE and also DML Exception handling. It has support for inline SOQL and SOSL query handling which returns the set of sObject records. We will study the sObject, SOQL, SOSL in detail in future chapters.

- Java like syntax and easy to use

  Apex is easy to use as it uses the syntax like Java. For example, variable declaration, loop syntax and conditional statements.

- Strongly Integrated With Data

  Apex is data focused and designed to execute multiple queries and DML statements together. It issues multiple transaction statements on Database.

- Strongly Typed

  Apex is a strongly typed language. It uses direct reference to schema objects like sObject and any invalid reference quickly fails if it is deleted or if is of wrong data type.

- Multitenant Environment

  Apex runs in a multitenant environment. Consequently, the Apex runtime engine is designed to guard closely against runaway code, preventing it from monopolizing shared resources. Any code that violates limits fails with easy-to-understand error messages.

- Upgrades Automatically

  Apex is upgraded as part of Salesforce releases. We don't have to upgrade it manually.

- Easy Testing

  Apex provides built-in support for unit test creation and execution, including test results that indicate how much code is covered, and which parts of your code can be more efficient.

When Should Developer Choose Apex?

Apex should be used when we are not able to implement the complex business functionality using the pre-built and existing out of the box functionalities. Below are the cases where we need to use apex over Salesforce configuration.

Apex Applications

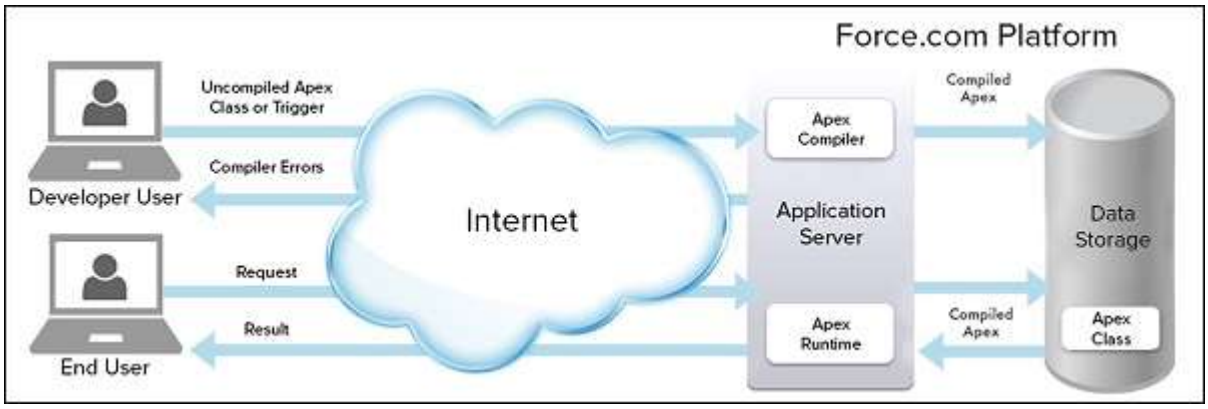
We can use Apex when we want to −

- Create Web services with integrating other systems.

- Create email services for email blast or email setup.
- Perform complex validation over multiple objects at the same time and also custom validation implementation.
- Create complex business processes that are not supported by existing workflow functionality or flows.
- Create custom transactional logic (logic that occurs over the entire transaction, not just with a single record or object) like using the Database methods for updating the records.
- Perform some logic when a record is modified or modify the related object's record when there is some event which has caused the trigger to fire.

Working Structure of Apex

As shown in the diagram below (Reference: Salesforce Developer Documentation), Apex runs entirely on demand Force.com Platform



Flow of Actions

There are two sequence of actions when the developer saves the code and when an end user performs some action which invokes the Apex code as shown below −

Developer Action

When a developer writes and saves Apex code to the platform, the platform application server first compiles the code into a set of instructions that can be understood by the Apex runtime interpreter, and then saves those instructions as metadata.

End User Action

When an end-user triggers the execution of Apex, by clicking a button or accessing a Visualforce page, the platform application server retrieves the compiled instructions from the metadata and sends them through the runtime interpreter before returning the result. The end-

user observes no differences in execution time as compared to the standard application platform request.

Since Apex is the proprietary language of Salesforce.com, it does not support some features which a general programming language does. Following are a few features which Apex does not support −

- It cannot show the elements in User Interface.
- You cannot change the standard SFDC provided functionality and also it is not possible to prevent the standard functionality execution.
- You cannot change the standard SFDC provided functionality and also it is not possible to prevent the standard functionality execution.
- Creating multiple threads is also not possible as we can do it in other languages.

Understanding the Apex Syntax

Apex code typically contains many things that we might be familiar with from other programming languages.

Variable Declaration

As strongly typed language, you must declare every variable with data type in Apex. As seen in the code below (screenshot below), lstAcc is declared with data type as List of Accounts.

SOQL Query

This will be used to fetch the data from Salesforce database. The query shown in screenshot below is fetching data from Account object.

Loop Statement

This loop statement is used for iterating over a list or iterating over a piece of code for a specified number of times. In the code shown in the screenshot below, iteration will be same as the number of records we have.

Flow Control Statement

The If statement is used for flow control in this code. Based on certain condition, it is decided whether to go for execution or to stop the execution of the particular piece of code. For example, in the code shown below, it is checking whether the list is empty or it contains records.

DML Statement

Performs the records insert, update, upsert, delete operation on the records in database. For example, the code given below helps in updating Accounts with new field value.

Apex Code Development Tools

In all the editions, we can use any of the following three tools to develop the code −

- Force.com Developer Console
- Force.com IDE
- Code Editor in the Salesforce User Interface

Conclusion:

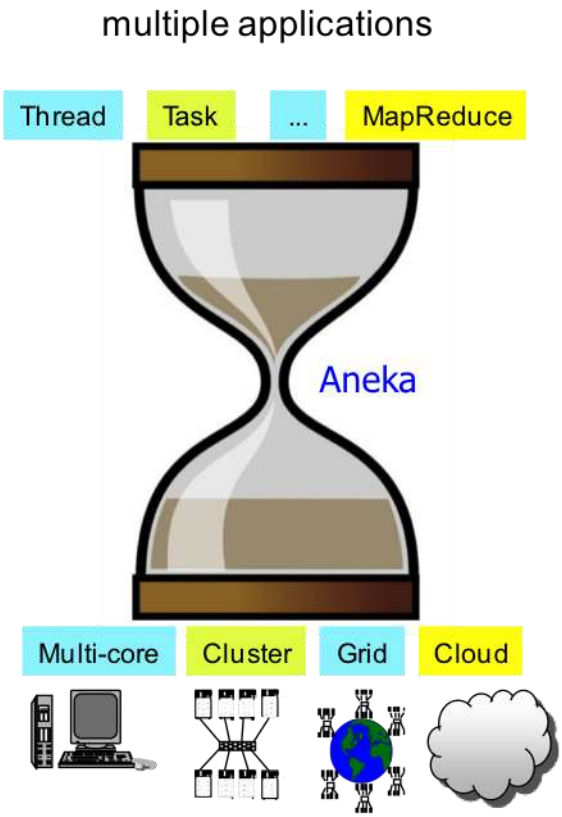
Reference: https://www.tutorialspoint.com/apex/apex_overview.html

Title:

Design an Assignment based on Working with Mangrasoft Aneka Software

Theory:

A Company named “Manjrasoft” is focused on the creation of innovative software technologies for simplifying the development and deployment of applications on private or public Clouds. The product Aneka plays the role of Application Platform as a Service for Cloud Computing. Aneka supports various programming models involving Task Programming, Thread Programming and MapReduce Programming and tools for rapid creation of applications and their seamless deployment on private or public Clouds to distribute applications.

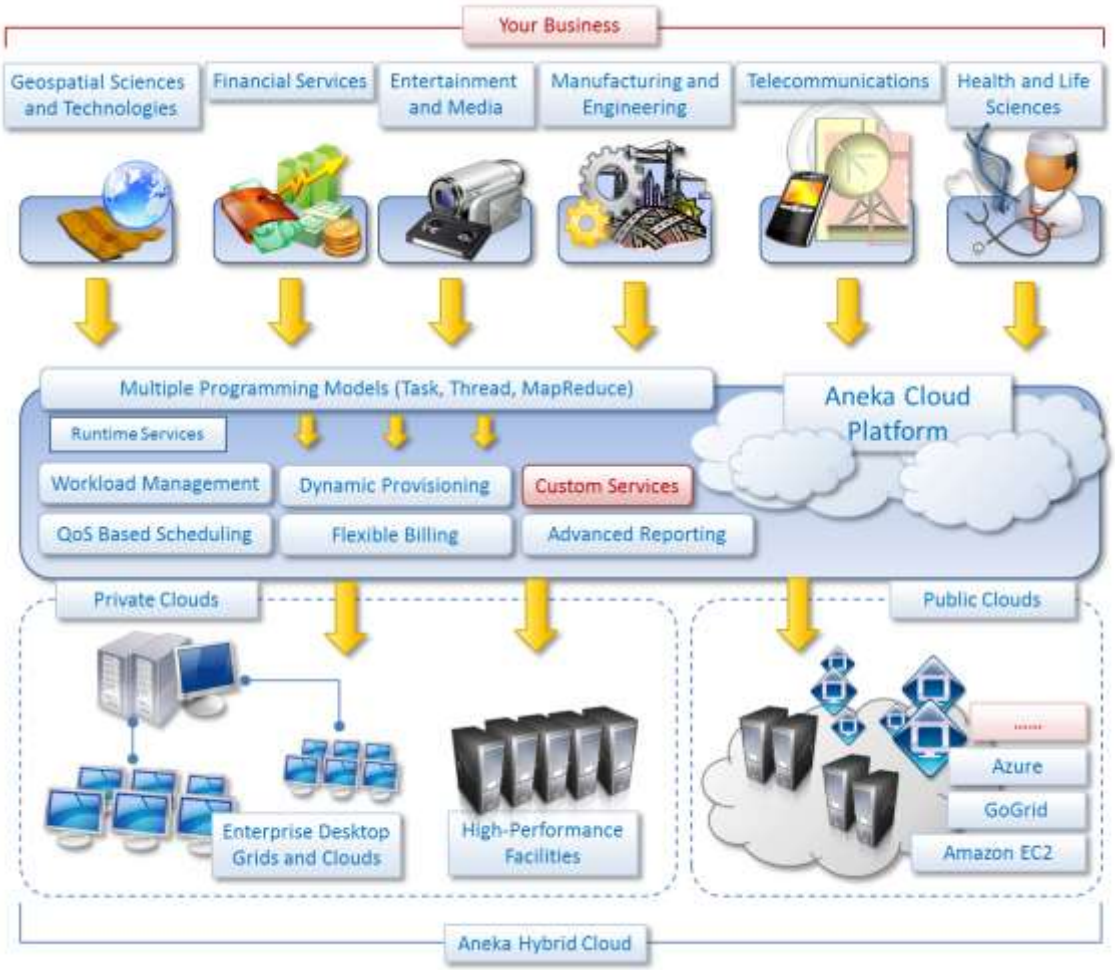


Aneka technology primarily consists of two key components:

1. SDK (Software Development Kit) containing application programming interfaces (APIs) and tools essential for rapid development of applications. Aneka APIs supports three popular Cloud programming models: Task, Thread, and MapReduce; and
2. A Runtime Engine and Platform for managing deployment and execution of applications on private or public Clouds.

One of the notable characteristics of Aneka PaaS is to support provisioning of private cloud resources ranging from desktops, clusters to virtual datacenters using VMWare, Citrix Zen server and public cloud resources such as Windows Azure, Amazon EC2, and GoGrid Cloud Service.

The potential of Aneka as a Platform as a Service has been successfully harnessed by its users and customers in three various sectors including engineering, life science, education, and business intelligence.



Highlights of Aneka

- Technical Value
- Support of multiple programming and application environments
- Simultaneous support of multiple run-time environments
- Rapid deployment tools and framework
- Simplicity in developing applications on Cloud
- Dynamic Scalability
- Ability to harness multiple virtual and/or physical machines for accelerating application result
- Provisioning based on QoS/SLA.

Business Value

- Improved reliability
- Simplicity
- Faster time to value
- Operational Agility
- Definite application performance enhancement
- Optimizing the capital expenditure and operational expenditure.

All these features make Aneka a winning solution for enterprise customers in the Platform-as-a-Service scenario.

Build

Aneka includes a Software Development Kit (SDK) which includes a combination of APIs and Tools to enable you to express your application. Aneka also allows you to build different run-time environments and build new applications.

Accelerate

Aneka supports Rapid Development and Deployment of Applications in Multiple Run-Time environments. Aneka uses physical machines as much as possible to achieve maximum utilization in local environment. As demand increases, Aneka provisions VMs via private clouds (Xen or VMWare) or Public Clouds (Amazon EC2).

Manage

Aneka Management includes a Graphical User Interface (GUI) and APIs to set-up, monitor, manage and maintain remote and global Aneka compute clouds. Aneka also has an accounting mechanism and manages priorities and scalability based on SLA/QoS which enables dynamic provisioning.

Conclusion:

Reference: http://www.manjrasoft.com/products.html