**AWS For DevOps**

What is Cloud Computing

[**https://aws.amazon.com/what-is-cloud-computing/**](https://aws.amazon.com/what-is-cloud-computing/)

Cloud computing is the on-demand delivery of IT resources over the Internet with pay-as-you-go pricing. Instead of buying, owning, and maintaining physical data centers and servers, you can access technology services, such as computing power, storage, and databases, on an as-needed basis from a cloud provider like Amazon Web Services (AWS).

Cloud Computing video from Azure

<https://www.youtube.com/watch?v=txZrgdehLaw>

Who is using cloud computing?

Organizations of every type, size, and industry are using the cloud for a wide variety of use cases, such as data backup, disaster recovery, email, virtual desktops, software development and testing, big data analytics, and customer-facing web applications. For example, healthcare companies are using the cloud to develop more personalized treatments for patients. Financial services companies are using the cloud to power real-time fraud detection and prevention. And video game makers are using the cloud to deliver online games to millions of players around the world.

# Benefits of cloud computing

* Agility
* Elasticity
* Cost saving
* Deploy globally in minutes

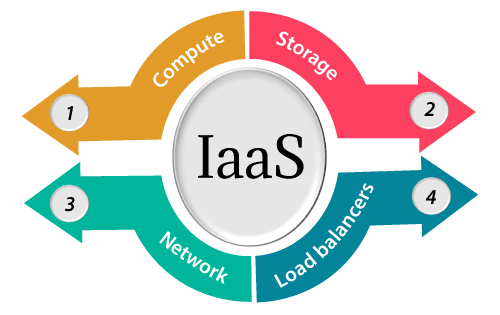
# Types of cloud computing

* Infrastructure as a Service (IAAS)
* Platform as a Service (PAAS)
* Software as a Service (SAAS)

**Infrastructure as a Service (Iaas)**

Iaas is also known as Hardware as a Service (HaaS). It is one of the layers of the cloud computing platform. It allows customers to outsource their IT infrastructures such as servers, networking, processing, storage, virtual machines, and other resources. Customers access these resources on the Internet using a pay-as-per use model.

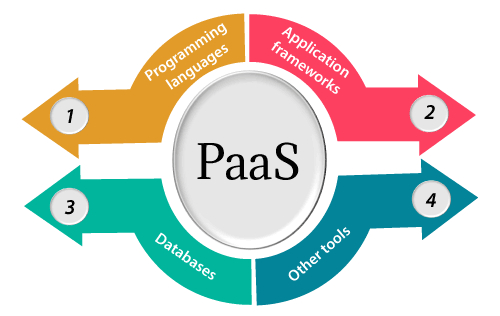
IaaS is offered in three models: public, private, and hybrid cloud. The private cloud implies that the infrastructure resides at the customer-premise. In the case of public cloud, it is located at the cloud computing platform vendor's data center, and the hybrid cloud is a combination of the two in which the customer selects the best of both public cloud or private cloud.



**Platform as a Service (PaaS)**

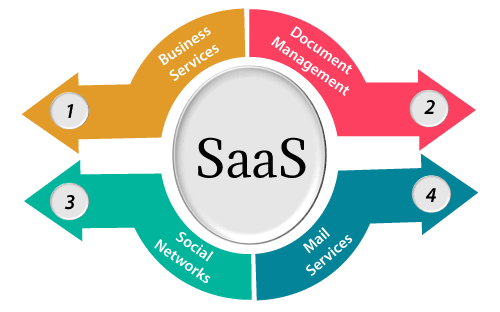
Platform as a service (PaaS) is a complete development and deployment environment in the cloud, with resources that enable you to deliver everything from simple cloud-based apps to sophisticated, cloud-enabled enterprise applications. You purchase the resources you need from a [cloud service provider](https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/choosing-a-cloud-service-provider/) on a pay-as-you-go basis and access them over a secure Internet connection.

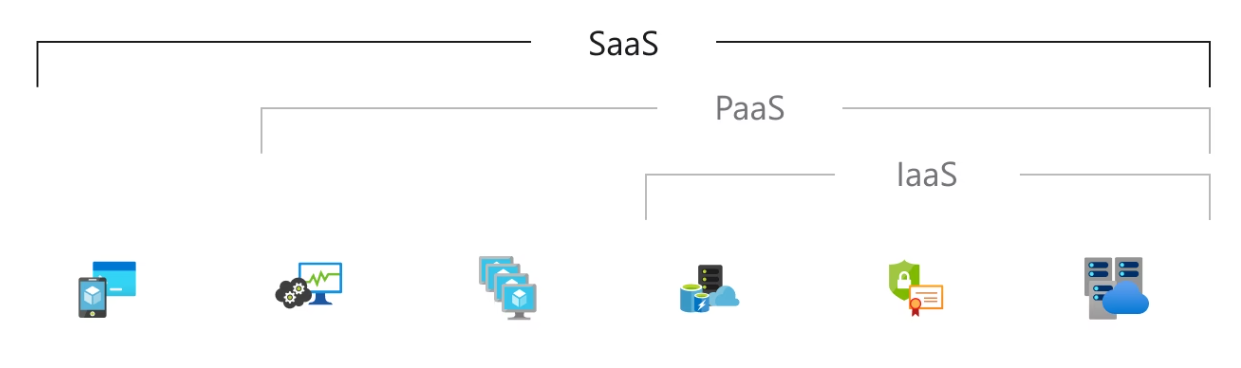
Like [IaaS](https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-iaas/), PaaS includes infrastructure—servers, storage, and networking—but also middleware, development tools, business intelligence (BI) services, database management systems, and more. PaaS is designed to support the complete web application lifecycle: building, testing, deploying, managing, and updating.



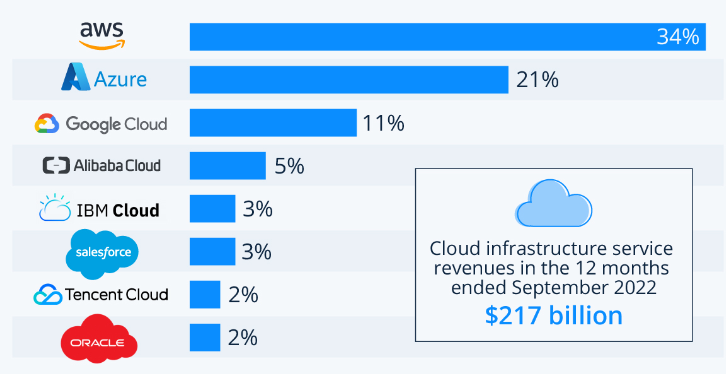
**Software as a Service (SaaS)**

Software as a service (SaaS) allows users to connect to and use cloud-based apps over the Internet. Common examples are email, calendaring, and office tools (such as Microsoft Office 365).





**TOP Cloud Providers**



**AWS Global Infrastructure**

The AWS Cloud spans **96** Availability Zones within **30** geographic regions around the world, with announced plans for **15** more Availability Zones and **5** more AWS Regions in Australia, Canada, Israel, New Zealand, and Thailand.

<https://aws.amazon.com/about-aws/global-infrastructure/>

**Regions, Availability Zones, Edge Locations**

**Regions**

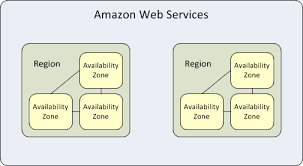
* **Regions are Large geographic areas**

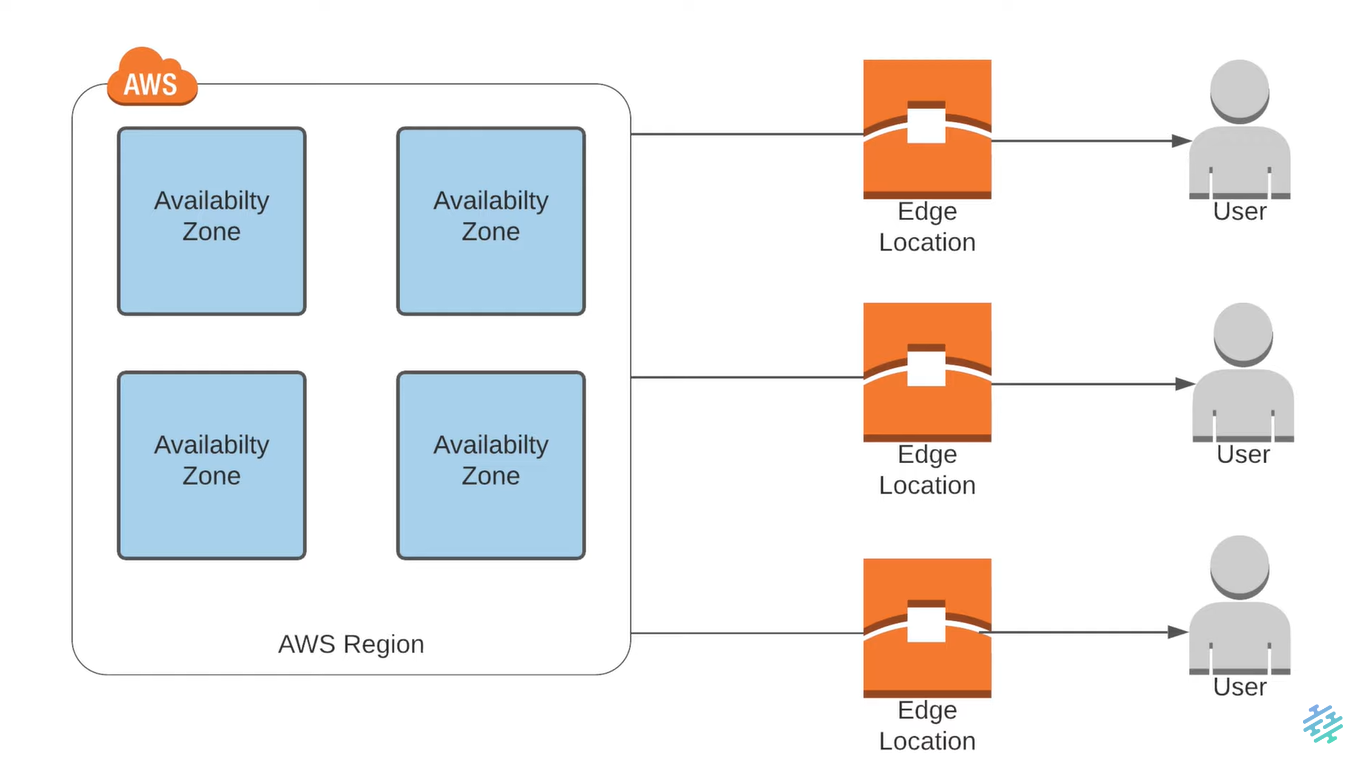
**Availability Zones**

* **Datacenters**

**Edge Locations**

* **Edge locations allows users to access content with lower latency**
* **A site that CloudFront uses to cache copies of your content for faster delivery to users at any location**





EC2 Instance

**Best Practices:**

* Gather the requirement
* Create the Key pairs
* Create the Security Groups
* Launch Instance

**Gather Requirement:**

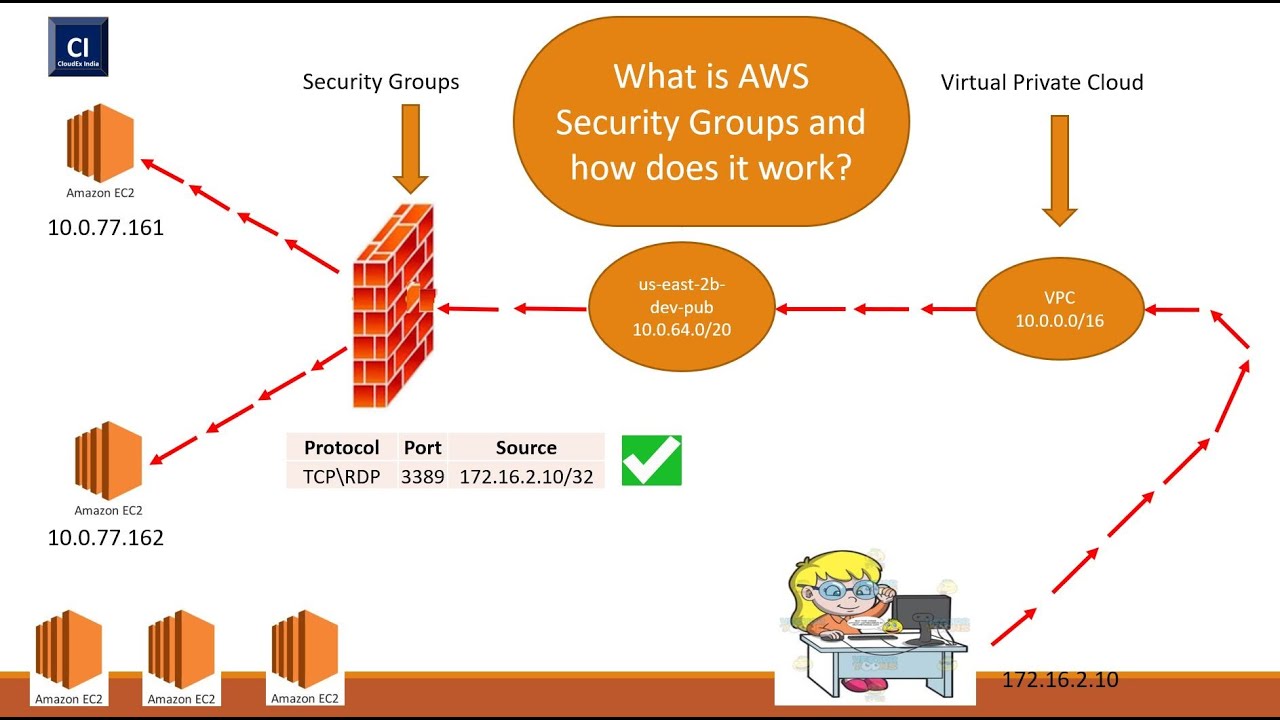
1. Operating System
   1. Ubuntu
   2. Centos
   3. Etc
2. Size of RAM, CPU, Network Etc
3. Storage size
   1. 10 Gig
4. Project Tag
5. Services/Apps will be Running
   1. SSH, HTTP, MySQL etc
6. Environment
   1. DEV, QA, Staging, Pre-Prod, Prod
7. Login user/owner

**Setup a Website on EC2 Instance**

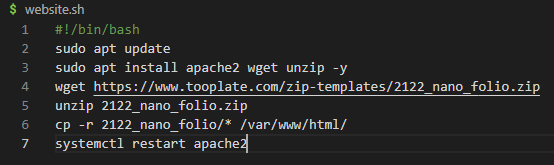
* Go to tooplate.com
* Create a Key Pair for this project – Give name as “Nano-Dev”
* Select .pem format & Click Create Key pair
* Private key will be downloaded into your system and public key will be stored in the AWS Key Pairs section
* Go to “Security Groups” 🡪 Create Security Group 🡪 Give name as “Nano-Dev-SG”

**Security Group**

* Security Group is nothing but a virtual Firewall which will allow / deny the incoming and outgoing traffic

****

* Launch Instance 🡪 Add Tags for Application & for Project
* Select the AMI
* Select the Instance-type
* Select the Key-pair which we created
* Select the Security Group which we created
* Login to EC2 instance from GIT Bash
* Copy & paste the website.sh content into a shell file like – website.sh



* Give it executable permission using **chmod +x website.sh** command
* Execute the shell script using “**sh website.sh or ./website.sh**” command
* Check the apache2 service using “**systemctl status apache2**”
* Allow port 80 from security group
* Now access the application using Browser

**More about EC2 Instance:**

* Stop and start the EC2 instance and check the public IP
* When you do this operation the Public IP will change. This is because AWS is assigning the Public IP from the pool. When you stop the Instance the IP will be released to the pool
* When you start the instance again, the public IP will be assigned again from the pool, but this time the IP will not be the same
* If you want to have fixed IP/Static IP then you have to use Elastic IP in AWS
* You will get 5 EIP by default by AWS, if you want more then you have to send request to the AWS support team and purchase more IP’s.
* Create a public IP, keep the same region name and click on Allocate.
  + Click on Actions
  + Click on “Associate Elastic IP address.
  + Choose the Instance and click on Associate
* Now Go to Instances
  + Select our Instance and Go to Actions tab
  + Settings
  + Change Instance Type
  + Before that you have to stop the Instance
  + Now you can change the instance type
* Network Interface
  + An elastic network interface is **a logical networking component in a VPC that represents a virtual network card.**
* Now go to Actions tab by selecting the Instance.
  + Select the Networking option
  + You will have options to Attach and Dettach to Network Interface
* Select Security in Actions tab
  + You can select different security group if you want to change the Security group

**AWS CLI**

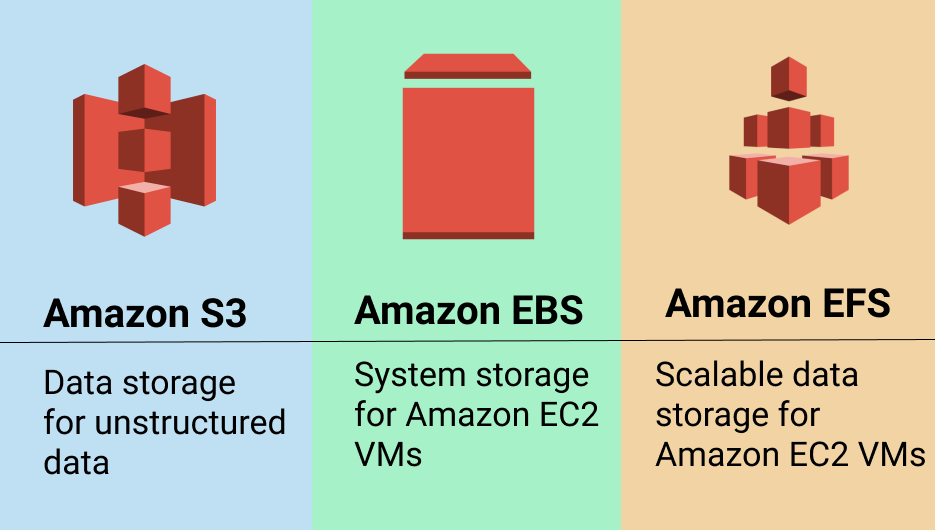
<https://aws.amazon.com/cli/>

* Before using the AWS CLI, you should be installing the AWS CLI in your laptop
* You can below command to install AWS CLI
  + Open Powershell as Administrator
  + $ choco install awscli -y
  + If you have not installed Chocolaty in your windows laptop, you can install it using below link
  + <https://chocolatey.org/install>
  + You can check which version of aws cli has been installed
    - $ aws –version
  + Configure the AWS CLI on the laptop
  + Create an IAM User in AWS
    - Give name as “awscli”
    - Select Programmatic Access
    - Click Next
    - Attach policy – select “AdministratorAccess”
    - Next and create it
    - Download the .csv file
    - Now go to GIT Bash on your laptop
    - Run below command to configure using create user
      * $ aws configure
      * Copy & Paste the Access key and secret access key
      * Type the region name: “us-east-1”
      * Output format: json
      * Now it will create a hidden directory like : /.aws/
        + ls ~/.aws/
        + cat ~/.aws/config
        + cat ~/.aws/credentials
      * Run below command to check the configuration details
        + $ aws sts get-caller-identity
      * Run some commands like
        + $ aws ec2 describe-instances

<https://docs.aws.amazon.com/cli/latest/reference/>

**EBS (Elastic Block Storage)**

* Block based storage
* Run EC2 instance OS, Stores data from DB, File Data etc.
* Placed in the same AZ of EC2 instance. Automatically replicated withing the AZ to protect from failure.
* Snapshot is backup of a volume



**Types of EBS**

* General Purpose (SSD)
  + Most work loads
* Provisioned IOPS
  + Large Databases
* Throughput Optimized HD
  + Big data and Data warehouses
* Cold HDD
  + File servers
* Magnetic
  + Backups & archives

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html>

**Create an EBS Volume and Attach it to EC2 instance**

* Go to Volumes section in EC2 Service
* Select Volumes
* Click on **Create volume**
* Select the Volume type
* Give the Volume size
* Choose the Availability zone for the EBS volume (Make sure to keep in the same AZ where the EC2 instance is running which you are going to attach)
* Give a Tag for the volume and click on Create volume
* Now to go to the volumes and select the newly created volume and click on Actions tab 🡪 Select “Attach volume”
* Select the Instance to which you want to attach the volume and click on Attach volume
* Now login to EC2 instance using SSH connection
* Go to /var/www/html
* There you will be having the img folder.

**Linux commands for Partition format and mount the new volume attached**

* $ fdisk -l
  + - This command will list all the disks available in the server
* $ df -h
  + - This show the disk space usage in Human readable format and where they are mounted
* $ fdisk /dev/xvdf
  + - This will open the fdisk utility for new volume called **“/dev/xvdf”**
    - Type **“m”** for all the options which you can use on this volume
    - We will use **“n add a new partition”**
    - Type **“n”**
    - Now Type **“p”** for primary partition
    - Now type the partition number like **“1”**
    - First sector 🡪 hit Enter button
    - Last sector 🡪 You can type like **“+3G”** it will take 3Gb from 5Gb or if you don’t type any number it will take entire 5Gb
    - Type **“p”** to print
    - Type **“w”** to write and partition will be created
* $ fdisk -l
* $ mkfs
  + - The **mkfs** command stands for**“make file system”** is utilized to make a file system
    - Hit the tab button two times it will display the available mkfs utilities
* $ mkfs.ext4 /dev/xvdf1
* Go to /var/www/html
* Create a backup directory 🡪 mkdir /tmp/img\_backup
* $ mv img/\* /tmp/img\_backup
* $ mount /dev/xvdf1 /var/www/html/img/ 🡪 This the temporary mount, if you reboot the server it be gone
* $ df -h
* $ umount /var/www/html/img/ 🡪 It will unmount the directory
* For Permanent mount you have to open a file called /etc/fstab
* $ vi /etc/fstab



* $ mount -a
* $ df -a

**EBS Snapshots**

* Firstly unmount the volumes which we have mounted
  + $ umount /var/www/html/img
  + $ df -h
* Detach the volume from the EC2 instance by going into volume section in AWS console
* Delete the volume
* Create a Volume 🡪 5gb
* Correct availability zone (AZ)
* Give name “db01-volume”
* Create volume
* Attach it to EC2 instance again
* Go to EC2 instance format and repeat the steps
* Create folder for mysql 🡪 mdkir /var/lib/mysql
* Go to fstab file and edit the mount path
* $ mount -a
* $ df -h
* $ apt install mariadb-server -y
* $ systemctl start mariadb
* $ systemctl status mariadb
* $ ls /var/lib/mysql/

**Snapshot Backup & Restore**

* Unmount partition
* Detach volume
* Create new volume from snapshot
* Attach the new volume created from snapshot
* Mount it back

**How to take the snapshot**

* Go to Volumes section in AWS console
* Select the volume and click on Actions
* Click on “Create snapshot”
* Give some description like “db01-volume-snapshot”
* Give a Tag
* Click on Create snapshot
* Check snapshot has been created

**How to recover the data from snapshot**

* Login to EC2 instance and delete the data from the directory which we have mounted like “/var/lib/mysql” using command
  + $ rm -rf \*
  + $ systemctl stop mariadb
  + $ umount /var/lib/mysql/
  + Now we have to detach the volume
    - Go to AWS console and go to Volumes
    - Select the volume and detach it
    - Now to Go to Snapshot in AWS
    - Select the snapshot and click on Actions
    - Select Create Volume
    - Give Tag name and click on Create Volume
    - Now go to Volumes section & you should see the recovered volume
    - Now go to Actions and attach the volume
    - Type command $mount -a
    - Now you will see all the data