Lecture 1

Cloud Computing Basics:

Characteristics,

Basic Architecture,

deployment models (IaaS etc.)

Lecture 2

Public cloud is an IT model where on-demand computing services and infrastructure are managed by a third-party provider and shared with multiple organizations using the public Internet. Public cloud service providers may offer cloud-based services such as infrastructure as a service (IaaS), platform as a service (PaaS), or software as a service (Saas) to users for either a monthly or pay-per-use fee, eliminating the need for users to host these services on site in their own data center.

Public cloud is a cloud deployment model where computing resources are owned and operated by a provider and shared across multiple tenants via the Internet.

Adv

## Cost Effectiveness

An extremely flexible pricing structure is one of the top benefits of the public cloud. Most of the public cloud providers give businesses the flexibility to pay by the hour. It helps businesses, especially the small and medium size, to tightly control their costs by paying for the infrastructure only based on their needs. Businesses can run their web applications without committing to a fixed cost for servers, software, setup or maintenance.

## Quick and Easy Set Up

Businesses can set up their public cloud within a matter of few hours. It can be easily bought on the Internet and deployed and configured remotely through the cloud provider website. Your IT team can very easily configure and manage the set up remotely with just an internet connection.

#### [**Download this free eBook to get a definitive guide to choosing the right public cloud provider. Get it Now!**](https://www.clariontech.com/cs/c/?cta_guid=2c3f777f-464b-49cc-b8e3-4132180f045a&signature=AAH58kH6Bx6WLZFcuaaA-_ryrH6JrJl0DA&pageId=5513905793&placement_guid=b335c092-4458-4799-a832-68000813ba5a&click=47b13e01-dbbd-4f6b-be55-7e18941bae31&hsutk=591d34a52ca68c4e0120bf1211f1def1&canon=https%3A%2F%2Fwww.clariontech.com%2Fblog%2F10-business-benefits-of-moving-to-public-cloud&utm_referrer=https%3A%2F%2Fwww.google.com%2F&portal_id=3852769&redirect_url=APefjpE9d-UfiRh-CiGhP2PK5KW9v6rbNPMH8W5jhwW9x2ze-h_Jk2t1_GKPTZF-EcmlhFojplGssqNExpr2pUwKHbOnCjfNcF2yW2PbVu-Ckns5jfVYDlZVyseFu7nWJJXMnCew_8uvhMLy7ZN71wdSrGOX53wbna5yzlRmhkQQuz36mXD9PchT0S6uaaph-oLm1UEY8xlCBHk_EU6zBtnW5OWaNOhSSQ&__hstc=156662500.591d34a52ca68c4e0120bf1211f1def1.1635006550154.1635006550154.1635006550154.1&__hssc=156662500.1.1635006550155&__hsfp=3176198904&contentType=blog-post)

## Optimization of Staffing Budgets

In many cases, the staffing budget makes up for more than half of the total computing costs – simply because good IT engineers are difficult to find and are expensive. With public cloud, companies need to budget only for the cloud services and since the management of cloud is extremely easy, they can always restructure their IT Teams and use the skillful resources in the areas that make more money for the business.

## No Maintenance

The cloud provider is responsible for the maintenance of the hardware, software, and networks in the cloud. Businesses, therefore, do not need to worry about keeping their infrastructure up-to-date or worry about aspects like security and upgrades. It allows them to run the infrastructure with a minimal IT staff, thereby significantly reducing the overall costs.

## No Long-Term Contracts

Small and medium businesses typically do not want to get into long term contracts and commit to a certain storage or bandwidth capacity because, a lot of times, they are not really sure about the real requirements. In such situations, the public cloud hosting works out very well because it does not require a long-term commitment or investment. The cloud providers usually offer pay-as-you-grow models, which make the overall engagement extremely easy and hassle-free.

## Economies of Scale

The public cloud offers massive economies of scale. It is extremely difficult to match it with private data centers. Businesses can be assured that the infrastructure is optimally used and the inevitable peaks and drops in the workloads are well taken care of. Since the infrastructure costs are shared across multiple users, the cloud providers typically optimize the hardware needs of its data centers and offer the services at lower costs.

## Agility

Today, businesses need to be quicker and more dynamic to be productive. They need to continuously evolve and improve their processes, tools, technologies, and policies. Being agile enables businesses to take quicker decisions and appropriately prioritize the work and ensure customer satisfaction. With public cloud, businesses experience simplified internal operations, better delivery, better collaboration, faster rollouts of new business initiatives, and improved data gathering and analysis ability.

## Global in Minutes

Be it the startups in India or online banks in New Zealand, several businesses which want to provide 24/7 access to their systems to their customers or employees are adopting the public cloud. Public cloud providers, with their massive networks of servers, network bandwidth, and the IT resources, are making the robust computing environment available and accessible for SMEs worldwide. In situations when a certain sensitive data needs to reside inside the boundaries of a particular nation, the public cloud makes it quick and easy to simply select the data center from that nation. The cloud providers have data centers across the world and businesses can select the one according to their business needs.

## High Flexibility without Redundancy

Before the era of cloud, businesses needed to buy additional hardware, storage, and software to prepare themselves for failure. This means that, in many cases, they had to duplicate their efforts and costs for ensuring the business continuity. With the cloud, the data is automatically mirrored on the data centers located at other locations – freeing up the businesses from worrying about data backup or excess costs.

## Maximum Uptime and Zero Risk Failure

Almost all the public cloud providers guarantee more than 99% uptime and no risk of failure. Since the overall cloud system interconnects several servers, in case of failure of any particular server, the other server takes over the workload automatically – ensuring a smooth and continuous performance for business critical applications.

Cloud computing is no more just a buzzword. It is a proven and widely used option by SMBs worldwide. Typically, applications which are web-based, customer-facing, and require multiple users to connect from different locations, are most suitable for cloud deployment.

Hybrid cloud architecture is the combination of public and private clouds by a wide area network or broadband connection, through which applications and data can be shared and which can be managed as a single IT architecture. Hybrid cloud infrastructure is well suited to fluctuations in demand for computing resources since it enables businesses to scale from on-premises to public cloud-based to meet increased demand, and scale back from the public cloud to on-premises (private cloud) only once demand recedes.

Many organizations utilize public cloud infrastructure as a service (IaaS) to process some workloads while retaining others in their private cloud, whether for cost, regulatory compliance, or technology reasons. The most common public IaaS providers are Amazon Web Services (AWS), Microsoft Azure and Google Cloud platform.

What's the difference between multi-cloud and hybrid cloud computing?

Both "multi-cloud" and "hybrid cloud" refer to cloud deployments that integrate more than one cloud. They differ in the kinds of cloud infrastructure they include.

A hybrid cloud infrastructure blends two or more different types of clouds, while multi-cloud blends different clouds of the same type. You might say hybrid cloud is like combining apples and oranges, while multi-cloud is like combining different types of apples.

What does multi-cloud mean?

"Multi-cloud" refers to the combination and integration of multiple public clouds. A business may use one public cloud as a database, one as PaaS, one for user authentication, and so on.

A cloud migration is when a company moves some or all of its data center capabilities into the cloud, usually to run on the cloud-based infrastructure provided by a cloud service provider such as AWS, Google Cloud, or Azure.

What are the Main Benefits of Migrating to the Cloud?Here are some of the benefits that compel organizations to migrate resources to the public cloud:Scalability - cloud computing can scale to support larger workloads and more users, much more easily than on-premises infrastructure. In traditional IT environments, companies had to purchase and set up physical servers, software licenses, storage and network equipment to scale up business services.Cost - cloud providers take over maintenance and upgrades, companies migrating to the cloud can spend significantly less on IT operations. They can devote more resources to innovation - developing new products or improving existing products.Performance - migrating to the cloud can improve performance and end-user experience. Applications and websites hosted in the cloud can easily scale to serve more users or higher throughput, and can run in geographical locations near to end-users, to reduce network latency.Digital experience - users can access cloud services and data from anywhere, whether they are employees or customers. This contributes to digital transformation, enables an improved experience for customers, and provides employees with modern, flexible tools.

A network access control list (ACL) is an optional layer of security for your VPC that acts as a firewall for controlling traffic in and out of one or more subnets. You might set up network ACLs with rules similar to your security groups in order to add an additional layer of security to your VPC.

3.

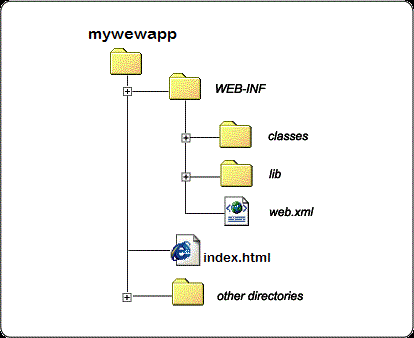
Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel.

4.

Directory Structure Of Java Web Applications

**Java Web Application Directory Layout**

In order to run your java web application(servlet,JSP etc) we need web server.Before running the application you need to package the resources inside it (servlets, JSP's,xml files etc.) in a standardized way as shown below.



**The Root Directory**

The root directory of you web application can have any name. In the above example the root directory name is mywebapp .

Under the root directory, you can put all files that should be accessible in your web application.

If your web application is mapped to the URL <http://localhost:8080/mywebapp/> then the index.html page will be accessible by the URL <http://localhost:8080/mywebapp/index.html>

If you create any subdirectories under the root directory, and place files in these subdirectories, these will be available by the subdirectory/file path, in the web application. For instance, if you create a subdirectory called pages, and put a file register.jsp inside it, then you could access that file from the outside, via this URL: <http://localhost:8080/mywebapp/pages/index.jsp>

**The WEB-INF Directory :**

The WEB-INF directory is located just below the web app root directory. This directory is a meta information directory. Files stored here are not supposed to be accessible from a browser (although your web app can access them internally, in your code).

Inside the WEB-INF directory there are two important directories (classes and lib, and one important file (web.xml). These are described below.

**web.xml :**

The web.xml file contains information about the web application, which is used by the Java web server / servlet container in order to properly deploy and execute the web application. For instance, the web.xml contains information about which servlets a web application should deploy, and what URL's they should be mapped to.

**classes Directory :**

The classes directory contains all compiled Java classes that are part of your web application. The classes should be located in a directory structure matching their package structure.

**lib folder :**

The lib directory contains all JAR files used by your web application. This directory most often contains any third party libraries that your application is using. You could, however, also put your own classes into a JAR file, and locate it here, rather than putting those classes in the classes directory.We will discuss more about this in next tutorial.

Instead of copying entire directory into server we can package them into war file and then do the deployment.

**What is war file :**

A WAR (or "web archive") file is simply a packaged webapp directory or archive file(like zip,rar etc) .It will store jsp,servlets,classes,meta data information,images,Sound and tag libararies etc. Its standard file extension is .war. WAR files are used to package Web modules. A WAR file is for a Web application deployed to a server. For example, an application for Stock trading might be distributed in an archive file called Stocktrading.war file.

The WAR file is a standard format for web applications that has specific directories and specific files. This includes a WEB-INF directory, a WEB-INF/web.xml file used to describe the application, a WEB-INF/lib directory for JAR files used by the application, and a WEB-INF/classes directory for class files that aren't distributed in a JAR. You would put the pages (JSPs and HTML) in the WAR as well. Then, you can distribute your application as one file, instead of as a collection of images, HTML pages, and Java classes.

**Note:-**Directory Structure of WAR file is exactly same as discussed above.In our next tutorial we will discuss how to develop and deploy war file on server.

Deploying a Web Service (Dir Structure)

# **Main Steps to Create and Configure a Web Application**

The following steps summarize the procedure for creating a Web application as part of an Enterprise application using the split development directory structure. See [Creating a Split Development Directory for an Application,](https://docs.oracle.com/cd/E13222_01/wls/docs90/programming/splitcreate.html) [Building the Applications,](https://docs.oracle.com/cd/E13222_01/wls/docs90/programming/splitbuild.html) and [Deploying the Application](https://docs.oracle.com/cd/E13222_01/wls/docs90/programming/splitdeploy.html) in Developing Applications with WebLogic Server.

You may want to use developer tools included with WebLogic Server for creating and configuring Web applications. See [Web Application Developer Tools](https://docs.oracle.com/cd/E13222_01/wls/docs90/webapp/basics.html#136762).

## Step One: Create the Enterprise Application Wrapper

1. Create a directory for your root EAR file:

\src\myEAR\

1. Set your environment as follows:
   * On Windows NT, execute the setWLSEnv.cmd command, located in the directory server\bin\, where server is the top-level directory in which WebLogic Server is installed.
   * On UNIX, execute the setWLSEnv.sh command, located in the directory server/bin/, where server is the top-level directory in which WebLogic Server is installed and domain refers to the name of your domain.
2. Package your Enterprise application in the \src\myEAR\ directory as follows:
   * Place the Enterprise applications descriptors (application.xml and weblogic-application.xml) in the META-INF\ directory. See Enterprise Application Deployment Descriptorsin [Developing Applications with WebLogic Server](https://docs.oracle.com/cd/E13222_01/wls/docs90/programming/index.html).
   * Edit the deployment descriptors as needed to fine-tune the behavior of your Enterprise application. See [Web Application Developer Tools](https://docs.oracle.com/cd/E13222_01/wls/docs90/webapp/basics.html#136762).
   * Place the Enterprise application .jar files in:

\src\myEAR\APP-INF\lib\

## Step Two: Create the Web Application

1. Create a directory for your Web application in the root of your EAR file:

\src\myEAR\myWebApp

1. Package your Web application in the \src\myEAR\myWebApp\ directory as follows:
   1. Place the Web application descriptors (web.xml and weblogic.xml) in the \src\myEAR\myWebApp\WEB-INF\ directory. See [weblogic.xml Deployment Descriptor Elements.](https://docs.oracle.com/cd/E13222_01/wls/docs90/webapp/weblogic_xml.html#1057399)
   2. Edit the deployment descriptors as needed to fine-tune the behavior of your Enterprise application. See [Web Application Developer Tools](https://docs.oracle.com/cd/E13222_01/wls/docs90/webapp/basics.html#136762).
   3. Place all HTML files, JSPs, images and any other files referenced by the Web application pages in the root of the Web application:

\src\myEAR\myWebApp\images\myimage.jpg

\src\myEAR\myWebApp\login.jsp

\src\myEAR\myWebApp\index.html

* 1. Place your Web application Java source files (servlets, tag libs, other classes referenced by servlets or tag libs) in:

\src\myEAR\myWebApp\WEB-INF\src\

## Step Three: Creating the build.xml File

Once you have set up your directory structure, you create the build.xml file using the weblogic.BuildXMLGen utility.

## Step Four: Execute the Split Development Directory Structure Ant Tasks

1. Execute the wlcompile Ant task to invoke the javac compiler. This compiles your Web application Java components into an output directory: /build/myEAR/WEB-INF/classes.
2. Execute wlappc Ant task to invoke the appc compiler. This compiles any JSPs and container-specific EJB classes for deployment.
3. Execute the wldeploy Ant task to deploy your Web application as part of an archived or exploded EAR to WebLogic Server.
4. If this is a production environment (rather than development), execute the wlpackage Ant task to package your Web application as part of an archived or exploded EAR.

**Note:**The wlpackage Ant task places compiled versions of your Java source files in the build directory. For example: /build/myEAR/myWebApp/classes.

How do you deploy the JAR/WAR/EAR files onto a VM? (Cloud)

# Deploying an App Engine application using an archive file



This page describes how to deploy your project to the App Engine flexible environment using a WAR archive or a runnable JAR file.

**Note:** The JAR file must specify a main method.

## Before You Begin

1. You need a Google Cloud project with an App Engine application to deploy to. If you don't already have one, use the Google Cloud Console to set up your Cloud project:

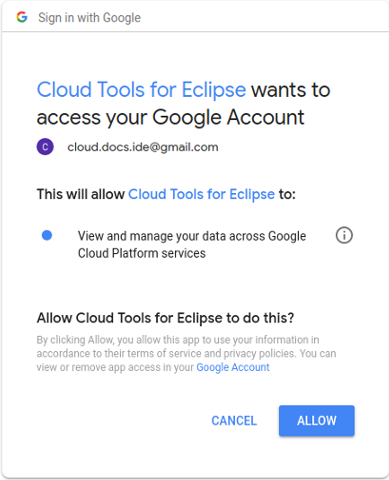
[Go to Cloud Console](https://console.cloud.google.com/projectselector/appengine/create?lang=java&st=true)

* 1. Select or create a new Cloud project.

1. Sign in to a Google account that is used to deploy your project to App Engine.
   1. Select **File** > **Sign in to Google**.

If you see **Manage Google Accounts** instead of the **Sign in to Google** option, that means you are already signed in, so you can skip these account sign in steps.

* 1. Your system browser opens outside of Eclipse and asks for the permissions it needs to manage your App Engine applications:



* 1. Click **Allow** and close the window. Eclipse is now signed into your account.

## Deploy a JAR or WAR file

To deploy a pre-existing WAR or runnable JAR file to the flexible environment:

1. Prepare an app.yaml file. At a minimum, it should contain the following lines:

runtime: java  
env: flex

1. Click the **Google Cloud** toolbar button .
2. Select **Deploy WAR/JAR File to App Engine Flexible...** in the drop-down menu.

A dialog box to configure the deployment. It provides a drop-down menu
 to select an Account, a list of Projects to deploy to, a field displaying
 the path to the WAR/JAR file, a button to Browse to a new WAR/JAR file, a
 field displaying the path to the app.yaml file, a button to Browse to a new
 app.yaml file, a checkbox to Promote the deployed version to receive all
 traffic, a checkbox to Stop previous version, a checkbox to include
 optional App Engine  configuration files, an expansion panel for
 Advanced options, and a field for entering a Staging bucket. 

1. Select the **Account** you want to deploy with, or sign in with a different account.
2. In the **Project** list box, select the Google Cloud project you want to deploy to.
3. In the **WAR/JAR** field, click **Browse** to select the archive file.
4. In the **app.yaml** field, click **Browse** to select your app.yaml file.
5. If you want to keep the current version running and manually promote the new version later using [gcloud](https://cloud.google.com/sdk) or the [Google Cloud Console](https://console.cloud.google.com/), clear the **Promote the deployed version to receive all traffic** checkbox.
6. If you don't want to stop the previous version, clear the **Stop previous version** checkbox.
7. Click **Deploy**.
8. After a successful deployment, Eclipse opens an internal browser connecting to the deployed app.

or

This article shows you how to deploy your code as a ZIP, WAR, JAR, or EAR package to [Azure App Service](https://docs.microsoft.com/en-us/azure/app-service/overview). It also shows how to deploy individual files to App Service, separate from your application package.

## Prerequisites

To complete the steps in this article, [create an App Service app](https://docs.microsoft.com/en-us/azure/app-service/), or use an app that you created for another tutorial.

If you don't have an [Azure subscription](https://docs.microsoft.com/en-us/azure/guides/developer/azure-developer-guide#understanding-accounts-subscriptions-and-billing), create a [free account](https://azure.microsoft.com/free/?ref=microsoft.com&utm_source=microsoft.com&utm_medium=docs&utm_campaign=visualstudio) before you begin.

## Create a project ZIP package

**Note**

If you downloaded the files in a ZIP package, extract the files first. For example, if you downloaded a ZIP package from GitHub, you cannot deploy that file as-is. GitHub adds additional nested directories, which do not work with App Service.

In a local terminal window, navigate to the root directory of your app project.

This directory should contain the entry file to your web app, such as index.html, index.php, and app.js. It can also contain package management files like project.json, composer.json, package.json, bower.json, and requirements.txt.

Unless you want App Service to run deployment automation for you, run all the build tasks (for example, npm, bower, gulp, composer, and pip) and make sure that you have all the files you need to run the app. This step is required if you want to [run your package directly](https://docs.microsoft.com/en-us/azure/app-service/deploy-run-package).

Create a ZIP archive of everything in your project. For dotnet projects, this folder is the output folder of the dotnet publish command. The following command uses the default tool in your terminal:

Copy

# Bash

zip -r <file-name>.zip .

# PowerShell

Compress-Archive -Path \* -DestinationPath <file-name>.zip

## Deploy a ZIP package

When you deploy a ZIP package, App Service unpacks its contents in the default path for your app (D:\home\site\wwwroot for Windows, /home/site/wwwroot for Linux).

This ZIP package deployment uses the same Kudu service that powers continuous integration-based deployments. Kudu supports the following functionality for ZIP package deployment:

* Deletion of files left over from a previous deployment.
* Option to turn on the default build process, which includes package restore.
* Deployment customization, including running deployment scripts.
* Deployment logs.
* A package size limit of 2048 MB.

5.6

How storage works

There are many different types of storage devices, and each use a seperate method, though they are similar.

Lets start with the HDD, or Hard Disk Drive. This is one that most people are familiar with. It uses a spinning magnetic-coated plate. Some larger drives have multiple plates, and most are dual-sided. There is a read-write head that reads the magnetic charges at each port on the plate. So a positive charge is a ‘1’ and a negative charge is a ‘0.’ When the plate is spinning really quickly, the read-write head can reach any part of the plate. The head reads in a string of numbers, i.e. 11010010001010011101001. And that is transfered and decoded into machine code in the processor via buses.

Now for a SSD, or Solid State Drive. These are a little harder to understand. Instead of using moving parts like the HDD, it uses a series of gate transistors. This is similar to how RAM works, but unlike the volatile properties of ram that uses DRAM, which are pools of transistors that require constant power to refresh the ‘’memory’, it uses NAND, which is non-volatile and does not need power to refresh the memory, keeping in mind it is still considered flash memory in both cases. Each gate has floating electrons with a positive or negative charge, and can be read accordingly. Because of the increased complexity and parts, these cost a lot more than HDD do.

A USB Flash Drive is very similar to a SSD. It uses transistors and flash memory that is stored in a non-volatile state that can be accessed via USB, or Universal Cereal Bus, through cables and motherboard.

Older honorable mentions:

CD, or Compact Disk, utilizes laser technology that shines a laser on the disk surface. The disk drives can read the amount of light reflected off of the grooves on the CD and detect what that groove represents, i.e. ‘x’ amount of light is represented as 0 and ‘y’ amount of light is represented as 1. When you ‘burn’ a CD, it creates these grooves on the surface that can be read through the disk drive later. Which you can imagine, after so many re-writes it wares down the CD pretty quickly.

Floppy Disk are very similar to cassette tapes. It uses a plastic based tape coated with iron oxide. This tape is ferromagnetic, meaning when it is exposed to magnetic fields, the properties are permanently magnetized by the field. You know the drill by now. Each section of magnetism can represent a 1 or a 0 and can be read accordingly.

NAS

Cloud Storage

Databases

NoSQL

Cloud DB

AWS SQL and NoSQL Datastores

Amazon Elastic Block Store (EBS)

Amazon Elastic Block Store (Amazon EBS) provides persistent block storage volumes for use with Amazon EC2 instances in the AWS Cloud. Each Amazon EBS volume is automatically replicated within its Availability Zone to protect you from component failure, offering high availability and durability. Amazon EBS volumes offer the consistent and low-latency performance needed to run your workloads. With Amazon EBS, you can scale your usage up or down within minutes—all while paying a low price for only what you provision.

Amazon Elastic File System (EFS)

Amazon Elastic File System (Amazon EFS) provides a simple, scalable, elastic file system for Linux-based workloads for use with AWS Cloud services and on-premises resources. It is built to scale on demand to petabytes without disrupting applications, growing and shrinking automatically as you add and remove files, so your applications have the storage they need – when they need it. It is designed to provide massively parallel shared access to thousands of Amazon EC2 instances, enabling your applications to achieve high levels of aggregate throughput and IOPS with consistent low latencies. Amazon EFS is a fully managed service that requires no changes to your existing applications and tools, providing access through a standard file system interface for seamless integration. Amazon EFS is a regional service storing data within and across multiple Availability Zones (AZs) for high availability and durability. You can access your file systems across AZs and AWS Regions and share files between thousands of Amazon EC2 instances and on-premises servers via AWS Direct Connect or AWS VPN.

Amazon EFS is well suited to support a broad spectrum of use cases from highly parallelized, scale-out workloads that require the highest possible throughput to single-threaded, latency-sensitive workloads. Use cases such as lift-and-shift enterprise applications, big data analytics, web serving and content management, application development and testing, media and entertainment workflows, database backups, and container storage.

Storage Gateway

The Storage Gateway is a hybrid storage service that enables your on-premises applications to seamlessly use AWS cloud storage. You can use the service for backup and archiving, disaster recovery, cloud data processing, storage tiering, and migration. Your applications connect to the service through a virtual machine or hardware gateway appliance using standard storage protocols, such as NFS, SMB and iSCSI. The gateway connects to AWS storage services, such as Amazon S3, S3 Glacier, and Amazon EBS, providing storage for files, volumes, and virtual tapes in AWS. The service includes a highly-optimized data transfer mechanism, with bandwidth management, automated network resilience, and efficient data transfer, along with a local cache for low-latency on-premises access to your most active data.

What is Amazon S3?

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. This means customers of all sizes and industries can use it to store and protect any amount of data for a range of use cases, such as data lakes, websites, cloud-native applications, backups, archive, machine learning, and analytics.

Use Cases

a. Build a Data Lake:Run big data analytics, artificial intelligence (AI), machine learning (ML), and high-performance computing (HPC) applications to unlock data insights.

b. Run Cloud-Native Applications :Build fast, powerful mobile and web-based cloud-native apps that scale automatically in a highly available configuration.

c. Backup and Restore Critical Data: Meet Recovery Time Objectives (RTO), Recovery Point Objectives (RPO), and compliance requirements with S3’s robust replication features.

d. Archive Data at Low Cost: Move on-premises archives to the low-cost S3 Glacier and S3 Glacier Deep Archive storage classes to eliminate operational complexities.