

Lecture Notes: Web Application on Cloud

1. Introduction to Web Application

1.1 Module Overview

In this module, you learnt about web applications, cloud, AWS and EC2.

Web Applications

Under web applications, you learnt about their basic concepts: What a web application is, how it works, the differences between a website and a web application, the three-tier structure of a web application, dominant web servers, such as the Apache HTTP and Tomcat servers, and various architectures of a web application.

Cloud

Under this topic, first, you learnt about some of the basic concepts of cloud computing, its working and the cloud-based architecture. You also learnt about the various cloud-based applications and cloud providers, and how they compare and stand against each other in terms of market share.

AWS and EC2

As part of this topic, first, you learnt a little more about AWS and saw how companies are benefiting from various cloud providers and why AWS is the leading cloud provider. Then you learnt about the benefits of EC2. Next, you learnt about a more practical approach and went through the common service offered by AWS: AWS EC2. Finally, you learnt about deployment and deployed web applications on the cloud, that is, on EC2.

1.2 Session Overview

In this session, you learnt what a web application is and how it works.

You also learnt about the benefits of using a web application, the differences between a website and a web application, and a few commonly used web applications such as upgrad, wikipedia etc.

In addition, you learnt about the three-tier of the web applications, its layers (Presentation layer, Logic layer and Data layer) and the benefits of using the three-tier architecture, and were also provided with a brief overview of the one-tier, two-tier and N-tier architectures.

You also learnt what a web server is, how it works and what is the need for a web server. Further, you learnt about the working of popular web servers, such as the Apache HTTP and Tomcat servers, and the differences between these web servers. Also, you were provided with a brief overview of other web servers, such as Windows IIS and Nginx.

Apart from this, you learnt about the various architectures of web applications (monolithic, Service-Oriented, microservices and serverless architectures) and were briefly introduced to microservice tools.

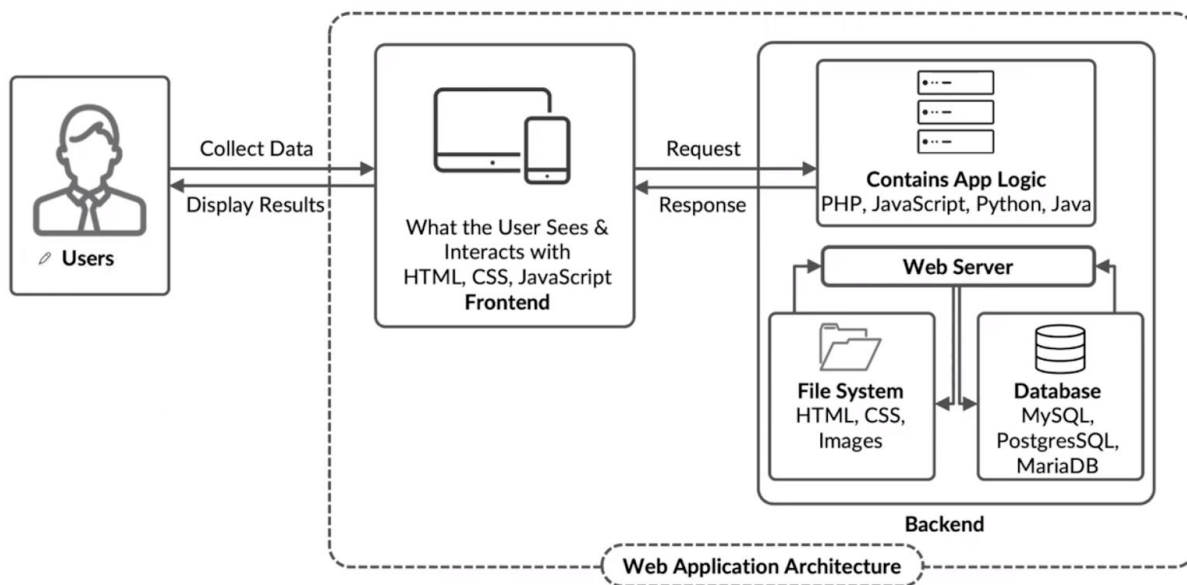
1.3 What is a Web Application?

Any application that actually runs on a web browser is called a web application. It does not depend upon the underlying operating system.

Examples of web applications include the Flipkart, BookMyShow etc.

As part of this topic, you learnt how a web application works. The user interacts with the front end and sends a request. After this, the front end interacts with the server to serve the user's request. You can have a look at the image given below to have a better understanding on working of web applications.

WORKING OF A WEB APPLICATION



Benefits of Web Applications

First, you learnt about the benefits of web applications, which include the following:

1. Independent of the operating system
2. Security
3. Reduced cost
4. Scalability
5. Easy maintenance
6. Increase in efficiency

Next, you learnt how a web application is important for a business.

1. It serves as a means of publicity and branding.

Web applications can immensely scale up to the popularity of the brand. It increases lead generation and builds the larger ground for sales with efficient communication with customers.
2. It provides options for enhanced customer support.

Web applications can be accessed anywhere and anytime. It serves as the first line of contact between businesses and potential customers, thus providing the best customer support.
3. It provides a competitive edge to the market.

Using a web application reduces the burden to maintain different applications, and by focusing on one application, companies can increase the uptime and enhance the customer experience.

You also learnt about the differences between a web application and a website, which are listed in this table.

	Web Application	Website
1.	It is designed for interaction with end users.	It is mostly static and accessible to all users.
2.	The users of a web application can read its content and also manipulate the data.	The users of a website can only read its content but cannot manipulate or affect its functioning.
3.	It follows the client-server architecture.	It is only server-based.
4.	It mostly requires the users to log in/sign up.	It rarely requires login/sign-up.
5.	Example: amazon.com, upgrad.com, etc.	Example: developer.mozilla.org, nodejs.com, etc.

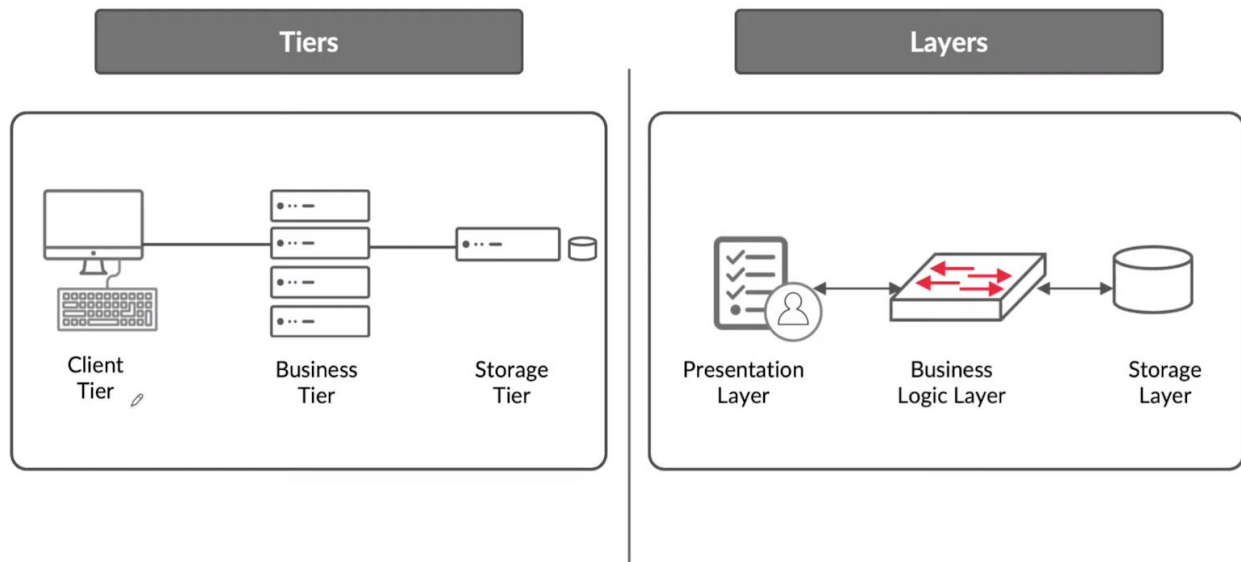
1.4 Web Application Architecture

An application can be one-tier, two-tier, three-tier or N-tier. Tiers are also known as layers. They can be categorised as:

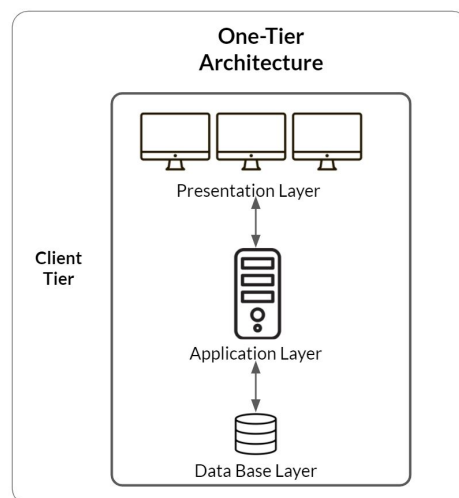
1. The presentation/client layer,
2. The application/business layer and
3. The database layer.

The presentation layer is used to view an application, while the business logic/application layer performs operations on the application and the storage/data layer is used to share/retrieve data. All the layers are independent of each other. They can, hence, be upgraded or replaced easily. The image below illustrates the web application architecture.

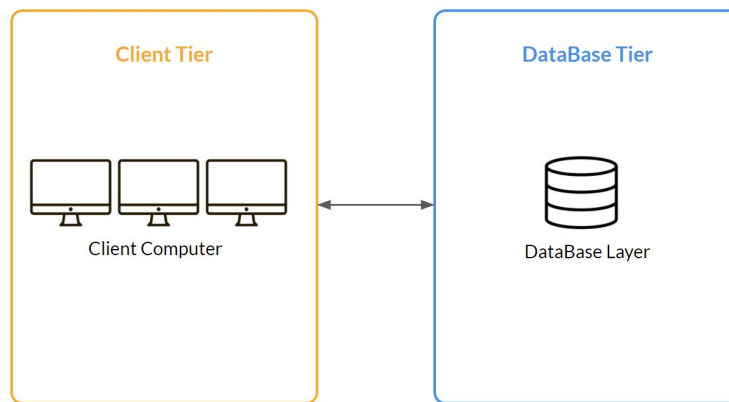
WEB APPLICATION ARCHITECTURE



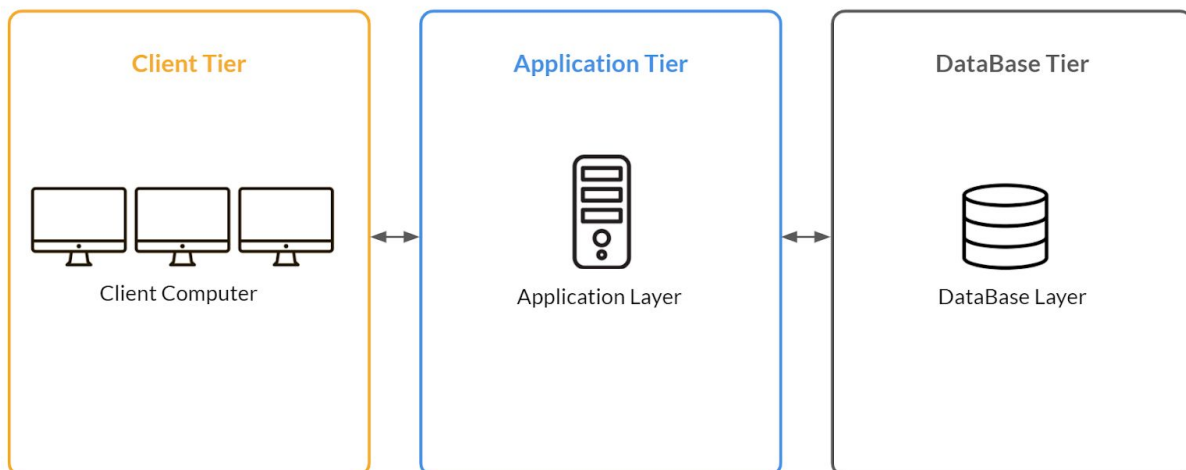
A one-tier application has all the layers in a single software package. These are also called standalone applications. This is illustrated in the image below.



A two-tier application has a client and a server. The client has the presentation layer and the business logic, while the database is at the server. All the queries from the client are solved at the database layer, which forms a bottleneck. The image below illustrates a two-tier application.



We can introduce one more layer in-between the two-tier architecture to make it a three-tier architecture. It is known as the business layer. All the queries are now handled by this layer and so, we can scale our business layer as per the requirement. The image below illustrates this architecture.



When any layer is broken into more than one layer to simplify the application or segregate any specific logic, it is known as the N-tier architecture. The N-tier system, also known as a distributed application, is an upgraded version of the three-tier system.

The three-tier architecture is the most popularly used architecture. The benefits of using this architecture are that it:

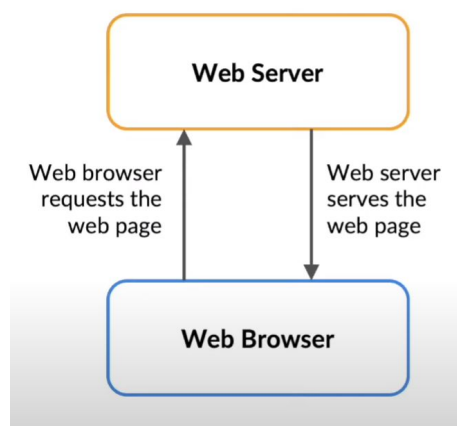
1. Increases the speed of development,
2. Provides scalability,

3. Offers flexibility and
4. Provides availability.

1.5 Web Servers - Apache HTTP and Tomcat Servers

Under this topic, first, you learnt that a web server is a software or hardware that can serve client requests over the internet. It is responsible for storing, processing and serving web pages. The communication between the client and the web servers takes place over the HTTP/HTTPS protocol.

Then you were explained, in simple terms, how a web server communicates with the web browser. This is illustrated in the image below.



Apache HTTP and Tomcat are the most commonly used web servers, and they are open source. The Apache HTTP server is used to serve static files, such as text, images, audio and video files, to the client, whereas the Apache Tomcat server delivers content that is dynamic, and which changes depending upon the client, i.e., who the client is, what the client has done on previous interactions with the server, etc.

The pros and cons of the Apache HTTP web server are mentioned in this image.



Pros

- Regularly updated with security updates
- Flexible in use as it has a modular structure
- Works with both Linux and Windows
- Due to a strong and huge community base, support is easily available

Cons

- Too many configurations can sometime be very overwhelming
- Under extremely high traffic load, can have performance degradation.

In the three-tier architecture, the Apache HTTP server lies in the client tier and the Tomcat in the middle tier. If a request made by a client is simple, then the Apache HTTP server handles the request and responds to the client. But if the request requires performing some logic, then it passes the request to the Tomcat server, which interacts with the database and sends the result to the Apache HTTP server, which further forwards it to the client.

The pros and cons of the Apache Tomcat web server are mentioned in this image.



Pros

- Light Weight
- Extra Layer of Security
- Very well Documented

Cons

- Not as fast as Apache to host static pages
- If not configured correctly, it can have memory leaks

Next, you learnt the following about Apache:

1. Communications occur mainly between the client and the server using the TCP/IP protocol.
2. HTTPS is served over port 443, whereas HTTP is served over port 80.
3. It is configured via the config files in which we have modules that control behaviour.
4. The default configuration allows Apache to listen to the IP address configured in the config files.
5. By default, Apache listens on port 80, but it can be bound to different ports for different domains as well.
6. Once a message reaches the destination, Apache shall send an ACK message. And in case a message is not delivered to the destination/client, it shall send a NAK message.

The differences between the Apache Tomcat and Apache HTTP servers are listed in this table.

	Apache Tomcat Server	Apache HTTP Server
1.	It is primarily used for Java applications.	It can be used with different programming languages, such as Perl, PHP, Python.
2.	It is less efficient for serving static web pages.	It is more efficient for serving static web pages.
3.	It is less configurable as compared with other web servers.	It is more configurable.
4.	The default port is 8080.	The default port is 80.

A reverse proxy is a server that sits in-between the client and the server and forwards requests from multiple clients to appropriate web servers. It is set up for increased security, increased scalability and flexibility.

1.6 Architectures of Web Applications

A web application consists of various components, such as a user interface, the business logic, a database, etc. The web architecture ensures that these components can work together flawlessly.

The various architectures of web applications include:

1. The monolithic architecture,
2. The Service-Oriented Architecture,
3. The microservices architecture and
4. The serverless architecture.

Monolithic Architecture

In this architecture, the entire code is written as a single block. All the features exist in a single codebase. It is used if the project is small and we do not have enough time.

The advantages and disadvantages of the monolithic architecture are listed in this table.

	Advantages	Disadvantages
1.	Ease in deployment	Maintenance
2.	Easy to test	Size of application can be a bottleneck
3.	Easy to scale horizontally	Need to redeploy the entire application in each update
4.	Easy to develop	Reliability
5.	Fast response time	Not easy to adapt to new technologies

Service-Oriented Architecture

In this architecture, the application is divided into multiple services, e.g. the front end, the back end, etc. The services are reusable, and interact with each other over the internet. IT is based on the principle of a 'service provider' and a 'service consumer'.

The advantages and disadvantages of the Service-Oriented Architecture are mentioned in this table.

	Advantages	Disadvantages
1.	Service reusability	High overhead

2.	Ease of maintenance	High investment
3.	Platform independence	Complex service management
4.	Availability and ease of deployment	
5.	Easy to scale	

1.7 Architectures of Web Applications - Microservices

Microservices Architecture

In this architecture, each service is divided into smaller independent services, which interact with each other to create the application. Each individual component interacts with another using APIs (Application Programming Interfaces). The principles of microservices are as follows:

1. Individual responsibility
2. Designed keeping failure in mind
3. Built around keeping new and available business capabilities in mind

The advantages and disadvantages of a microservices architecture are mentioned in this table.

	Advantages	Disadvantages
1.	Microservices are self-contained, independent deployment modules.	Microservices have all the associated complexities of the distributed system.
2.	The cost of scaling is relatively less than that of the monolithic architecture.	There is a higher possibility of failure throughout communication between totally different services.
3.	Microservices are independently manageable services. They can enable more and more services as the need arises. It minimises the impact on an existing service.	It is difficult to manage a higher number of services.
4.	It is possible to vary or upgrade each service individually, instead of upgrading within the entire application.	Network latency and load balancing are two issues.

5.	Microservices permit us to develop an application that is organic in nature (an application that latterly upgrades by adding more functions or modules).	Complex testing is done over a distributed environment.
6.	They enable event streaming technology to allow easy integration as compared with heavyweight interposes communication.	The cost associated is high.
7.	Microservices follow the single responsibility principle.	
8.	The service in demand is often deployed on multiple servers to improve performance.	
9.	There is less dependency and they are easy to test.	
10.	You can scale as per your need.	
11.	The release cycle of the application gets faster.	

Previously, you learnt about the different microservices. Now, the table below lists the differences between the monolithic and microservices architectures.

DIFFERENCE

Key	Monolithic architecture	Microservices architecture
Basic	Monolithic architecture is built as one large system and is typically one code-base	Microservices architecture is made up of small independent module based on business functionality
Scale	Based on the increase in demand, it can not be scaled easily.	Based on the increase in demand, it can be scaled easily.
Database	It has shared database	In most of the cases, the modules has their own database
Deployment	A single and massive code base makes development slow and time consuming.	Each project is independent and small in size. So overall time for build and development is less.
Tightly Coupled and Loosely coupled	It extremely difficult to vary technology or language or framework because everything is tightly coupled and depend on each other	Easy to vary technology or framework as every module and project is independent

Next, you saw the use case of Uber. You saw how the company moved to the microservices architecture from the monolithic architecture, which made its expansion easier.

We previously discussed APIs using which the services in the microservices architecture interact with each other. REST APIs are a set of rules, which allow the services to talk to each other. APIs are stateless in nature and communicate over the HTTP protocol.

The API gateway is the entry point for all the APIs.

1.8 Session Summary

You have successfully completed the first session of this module. Now, we will quickly review what you have learnt in this session.

In this session, we covered these topics:

- What is a web application? How does it work?
- Differences between a website and a web application
- The one-tier, two-tier, three-tier and N-tier architectures of web applications
- What is a web server? Why do we need it? How does it work?
- The Apache HTTP and Tomcat servers
- The monolithic, Service-Oriented and microservices architectures

2. Introduction to Cloud

2.1 Session Overview

In this session, first, you learnt about the concepts of cloud computing, which refers to the on-demand delivery of IT services over the internet. Then we discussed various cloud-based applications. In this session, you learnt these topics:

- Introduction to cloud computing
- Benefits of using cloud
- Cloud-based architecture
- Cloud deployment and service models

2.2 Introduction to Cloud Computing

Today, cloud computing is prevalent in a wide range of applications, such as Alexa, Siri, WhatsApp, Gmail, Microsoft Office. It plays a major role in the IT industry.

You learnt through an example the different scenarios of how cloud computing works.

The process of delivering IT services to users and organisations over the internet based on their demands is known as cloud computing. It refers to the delivery of services such as processing, storage, databases, networking, etc. to users and organisations over the internet based on their requirements. The servers on which these software and databases run are located in data centres across the world. Users and organisations can access these servers through the internet from anywhere.

Cloud computing is a pay-as-you-go service; this means the users pay only for the services that they use.

The two main users of clouds are:

- **End users**, who use the cloud services for proprietary benefits, and
- **Business management users**, who utilise the cloud services at an organisational level.

Next, you saw the use case of Netflix. You saw how the company moved to cloud to scale for a large user base and increase its reach.

The three major cloud providers are:

- Amazon Web Services (AWS),
- Microsoft Azure and
- The Google Cloud Platform.

2.3 Benefits of Cloud Computing

As part of this topic, you learnt about the benefits of using clouds over traditional data centres. These are the benefits:

- **Cost-saving:** Cloud computing eliminates the need for buying or maintaining hardware and software resources and setting up data centres. It also eliminates the need for maintaining computing infrastructure. Most of the cloud services are pay-as-you-go, which means customers only pay for the services they use. The use of a cloud also reduces the cost associated with staff wages.
- **Scalability:** Cloud computing provides businesses with the ability to expand their resources when needed. Organisations need not worry about installing infrastructure, as it is done by the cloud service providers. They can also scale up their applications as per requirement.
- **Availability:** Cloud is available at different locations across the world. It allows companies to expand to new geographical regions and deploy the resources globally.
- **Security:** Cloud allows access to data and applications to authorised and authenticated users only.
- **Data storage space:** Organisations can opt for the exact amount of storage they need and pay only for the space that they use.

2.4 Cloud-Based Architecture

Cloud-Based Architecture

A cloud-based application has an architecture, which contains two basic components: The front end and the back end. Both of these components are connected to each other through the internet.

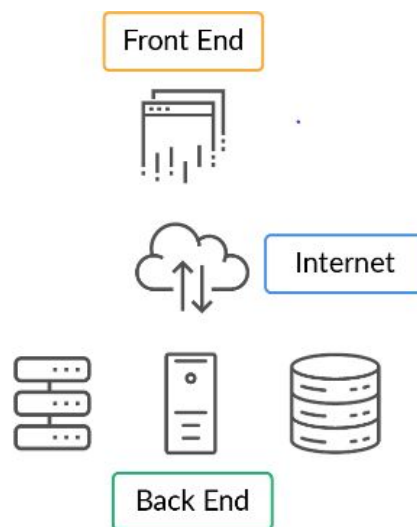
Cloud architecture is a combination of different technologies, which are brought together to create the cloud. This provides shared scalable resources across a network.

Cloud architecture has these two components:

- **Front end:** This is the client or the end user. It consists of all the applications and interfaces that are used by the clients to access cloud resources.
- **Back end:** It is the cloud itself, as it consists of infrastructure such as databases, computing resources, deployment models, which are required to build the cloud.

The front end and the back end are connected to each other through a network, usually the internet.

This image illustrates the schematic structure of a cloud-based architecture.



2.5 Session Summary

In this session, you learnt what cloud computing is and what are the applications that use cloud computing. You learnt about the benefits of cloud computing, such as cost-saving, scalability, availability, security.

You learnt about the cloud architecture and its different components. You also learnt about the three main types of cloud deployment models: Private, public and hybrid.

Finally, in the optional part, you learnt about cloud services and the benefits that they offer to users and organisations. The three cloud service models are Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).

3. Introduction to AWS and EC2

3.1 Session Overview

In this session, you learnt about the services provided by a popular cloud platform, Amazon Web Services. You performed the steps for launching an Amazon EC2 instance and then logged into it to access the instance. At the end of this session, you deployed the UPSTAC application to the instance.

3.2 Introduction to AWS

Amazon Web Services (AWS) provides you with managed services on its cloud platform. It manages software installation and hardware maintenance and provides the necessary resources to users for running big data systems, thereby freeing them from the burden of managing any hardware, software, network, etc. The users may then perform a one-time set-up process, beyond which everything is managed by AWS.

The Amazon EC2 of AWS provides computing resources with complete control over them. In the segments that follow, you used Amazon EC2 (virtual server) to deploy a web application on the Apache HTTP server.

3.3 Introduction to NuvePro

In this segment, you learnt about the *NuvePro* dashboard and set up an EC2 instance on your Amazon AWS account. You also learnt about a few best practices

that should be followed while using EC2. It is always important to follow these best practices as they will prevent you from spending unnecessary AWS credits.

Notes on the NuvePro Lab

While you are working on your lab, there are certain critical pointers that we would like you to keep in mind:

- It may take some time (up to 3 hours) for the activation of your AWS account after clicking the Jump to Console button for the first time. Please get in touch with your student mentor if you are unable to access your AWS dashboard even after 3 hours.
- As a general practice, please shut down your EC2 instances when you are not using them.
- Your usage bill information is displayed on the left panel of your NuvePro dashboard. Monitor this closely.
- At any given time, you will be assigned a “budget value”. This value can only be increased in certain steps. You can find out this value from your student mentor. From this budget value, subtract your usage to find how much credit you have remaining.
- Please add the security group of **MyIP** to your AWS lab before you start any work on your AWS cluster.
- Please note that the billing of an EC2 instance happens on an hourly basis. So, do not start-stop your instance multiple times. As soon as you start your EC2 instance, you will be charged the minimum amount, which is the per hour amount, even if you stop your instance within another minute. If you now start it again, then the charge of 1 hour will be deducted from your budget. Hence, we advise you to work with your instance for longer stretches (at least 1 hour) or avoid starting and stopping the instance multiple times (at least not within an hour).

This [document](#) contains all the steps to log in to your NuvePro dashboard.

3.4 Virtual Machine on Cloud

In the previous segment, you learnt about the services provided by AWS and learnt how to access the AWS console.

In this segment, you learnt about the steps to create a Virtual Private Cloud (VPC) and security groups. You also learnt how to launch a t2.micro EC2 instance.

Here, the t2.micro type instance was launched for your practice, and it should be terminated as per the instruction. Later in the session, you launched an m4.xlarge instance, which was used primarily for deploying your web-based application.

Amazon Virtual Private Cloud

The Amazon Virtual Private Cloud (VPC) allows users to provision an isolated section of AWS according to their requirements. Users can utilise the resources of AWS in a virtual network. They can also customise their virtual environment according to their needs.

Note: The VPC set-up is a one-time process, and you can make use of the VPC created here in the instances that you will launch in the future.

Security Groups

There is a security group associated with each EC2 instance. It provides security at the port and protocol access levels. Each security group acts as a firewall for the instance, and it has a defined set of rules that filter the traffic coming to and going out of the instance. A security group mainly consists of two types of rules: Inbound and outbound. The inbound rules define the traffic coming to the EC2 instance. The outbound rules define the traffic going out of the EC2 instance. The VPC uses security groups to control the traffic at the instance level.

MyIP Set-Up

In all your EC2 instances, you will set your **inbound rules** as **All TCP** and mention the source as **MyIP** to allow only your machine to access the EC2 instance. Since the IP that the MyIP feature of the AWS security group takes is the public IP of your machine, which is subject to change depending upon your internet service provider, we instruct you to follow this step every time you want to run your EC2 instance.

You configured the EC2 instance with the help of these documents:

[Steps to Change the AWS Console Interface](#)

[Steps to Create a VPC and Security Groups](#)

[Steps to Launch the EC2-t2.micro Instance](#)

3.5 EC2 Log-In and File Transfer

In this segment, you learnt how to access the EC2 instance by logging in to it. You also learnt how to access the EC2 instance on Windows as well as Linux/Mac systems. In addition to this, you learnt the steps to transfer files between the local system and the EC2 instance.

To access the EC2 instance from a Windows system, we use PuTTY.

PuTTY

PuTTY is a serial console and a network file transfer application. It allows users to connect to a remote system from their host machine. You used PuTTY to log in to your EC2 instance.

To transfer files from the local system running on Windows to the EC2 instance, and vice versa, you use WinSCP.

WinSCP

WinSCP (Windows Secure Copy) is an open-source file transfer client for Windows. It is used for transferring files from the local computer to the remote machine. It protects private information, such as login ID and password, by using cryptographic methods. Here, you learnt how to transfer files from the EC2 instance to the local machine, and vice versa, using WinSCP.

Mac and Linux users can simply use the SSH command from their terminals to log in to the EC2 instance, and then use the SCP command to transfer files. Here, you learnt how to transfer files from the EC2 instance to the local machine, and vice versa, using the terminal.

You used different documents to perform the steps above. Here are the links to those documents:

[EC2 Log-In - Windows](#)

[EC2 Log-In - Linux/Mac](#)

[MyIP Set-Up](#)

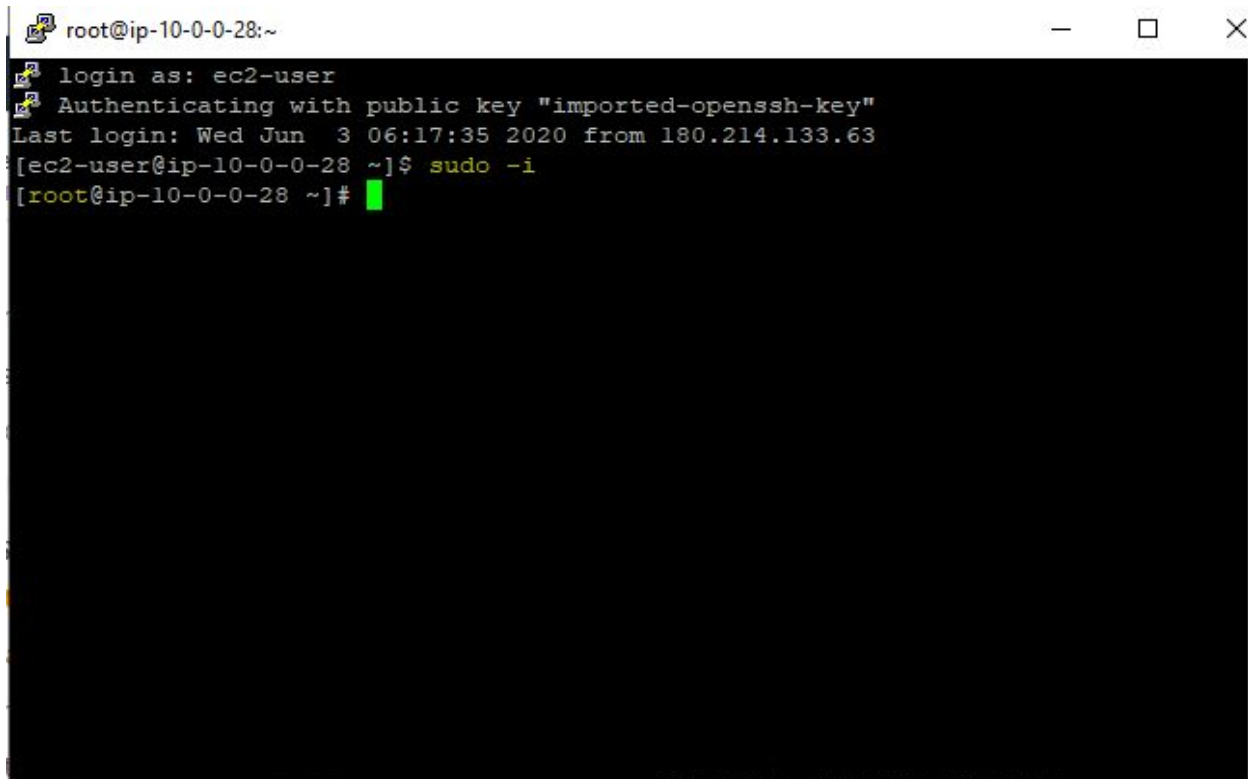
[File Transfer - Windows](#)

[File Transfer - Linux/Mac](#)

3.6 Practising Linux Commands

In this segment, first you learnt about some basic shell commands, which would help you throughout the program.

Then you tried to execute the commands provided in [this](#) document after going to the root user using the command **sudo -i** as shown in this image.

A terminal window titled 'root@ip-10-0-0-28:~' with standard window controls. The terminal shows an SSH login for 'ec2-user' using a public key. It displays the last login time and IP address. The user then runs 'sudo -i' to become root, and the prompt changes from '\$' to '#'.

```
root@ip-10-0-0-28:~  
login as: ec2-user  
Authenticating with public key "imported-openssh-key"  
Last login: Wed Jun  3 06:17:35 2020 from 180.214.133.63  
[ec2-user@ip-10-0-0-28 ~]$ sudo -i  
[root@ip-10-0-0-28 ~]#
```

Next, you learnt about the **vi** editor by going through this [link](#). This editor is present in Unix systems and you will be using throughout the course.

Then you learnt how to stop an EC2 instance successfully to save the cost associated with it. The steps for this are mentioned in [this](#) document.

3.7 EC2 Instance Termination

Even if your EC2 instance is in the stopped state, there is still a fixed cost associated with it. And since you had finished practising with the t2.micro instance in this segment, you learnt about the steps to terminate an EC2 instance.

The steps for this are mentioned in [this](#) document.

3.8 Demonstration of the Apache Server on EC2

In this segment, you learnt how to create an EC2 instance. You also deployed your Apache server on the EC2 instance.

3.9 Demonstration on Deploying the UPSTAC Application on EC2

In this segment, you learnt how to deploy a web application on the cloud. You deployed an application named UPSTAC (Unified Platform for Syndromic mapping, Testing, Analytics and Consultation), which is essentially a simple healthcare-based web application. You learnt more about the UPSTAC application and its user interface, and also saw use cases and took a brief look at its working.

Then you learnt how to deploy the UPSTAC application on the EC2 instance step by step.

3.10 Session Summary

In this session, you learnt how to use the Amazon cloud with the help of a step-by-step guide. Here is the summary of your learnings from this session:

- Introduction to AWS and the services that it provides
- Understanding the NuvePro dashboard and accessing the AWS console through it
- Launching the t2.micro EC2 instance and logging into the instance using PuTTY in Windows and the SSH command in Linux/Mac. Also, transferring files using WinSCP in Windows

Miscellaneous

1. Serverless Architecture

In the previous session on the '**Architecture of Web Applications**', you learnt about the three basic architectures: The monolithic architecture, the Service-Oriented

Architecture and the microservices architecture. Apart from these architectures, there is another architecture, called the serverless architecture.

The serverless architecture does not mean 'no server'. Rather, it means that the developer can simply deploy the code and the code execution is taken care of by a third-party cloud service provider, such as AWS. Thus, the developer need not worry about server maintenance and provision.

When the demand for your application is rising rapidly, you may feel a need to expand the application's power and accessibility. You can add more power to your existing machine with vertical scaling, and with horizontal scaling, you can get additional resources into your system by adding more machines to your network, thus sharing the processing and memory workload across multiple devices.

2. Cloud-Based Deployment Models

Cloud Deployment Models

The three major types of cloud deployment models available include private cloud, public cloud and hybrid cloud.

1. Private Cloud

In a private cloud, the cloud resources are operated solely for a single organisation. The cloud is managed either by the organisation itself or by a third party and is hosted internally or externally.

Some of the private cloud providers include:

- IBM,
- Oracle,
- VMware and
- The Hewlett Packard Enterprise (HPE).

Here are some of the advantages and disadvantages of a private cloud.

Advantages

- It provides high security and also restricts access to authorised users only. Hence, this kind of infrastructure is generally preferred in financial institutions, such as banks, insurance firms.
- It provides high control over the resources.

Disadvantages

- It is not cost-effective when compared with a public cloud.
- It has limited scalability and can be scaled only up to the internal hosted resources.

2. Public Cloud

In a public cloud, the cloud resources are owned and operated by a third party. The services are provided to the users via the internet. The cloud service provider is solely responsible for maintaining the resources.

Following are some of the major public cloud service providers:

- Google Cloud Platform
- Microsoft Azure
- Amazon Web Services (AWS)

Here are some of the advantages and disadvantages of a public cloud.

Advantages

- It is highly scalable. It offers flexibility to either scale up or scale down the usage of resources as per the demand or based on a user's request.
- It is also cost-effective. The users of a public cloud have to pay for only what they use.

Disadvantages

- A public cloud might have security issues.
- It is not 100% customisable as per an organisation's requirements.

3. Hybrid Cloud

A hybrid cloud is a combination of a public and a private cloud, and it allows organisations to share data.

It can be a combination of:

- At least one private cloud and at least one public cloud,
- Two or more private clouds, or
- Two or more public clouds.

The clouds chosen for the development of hybrid clouds should be able to connect multiple computers through a single network. These clouds should also be able to

move workloads between one environment and another. The ordered or consistent connections between separate clouds make it a hybrid cloud. Usually, the performance of a hybrid cloud is dependent upon the development and management of its connections. A public and a private cloud are linked using complex networks of LANs, APIs, VPNs, etc.

Several cloud providers offer their customers preconfigured connections; for example:

- Dedicated Interconnect by Google Cloud,
- Direct Connect by AWS and
- ExpressRoute by Microsoft Azure.

Here are some of the advantages and disadvantages of a hybrid cloud.

Advantages

- A private cloud is secure, and, hence, a hybrid cloud is secure as well.
- It offers scalability. You already know that a public cloud is scalable. Therefore, a hybrid cloud, which is a combination of a public and a private cloud, is also scalable.
- Users can access both the private and the public cloud as per their requirements; thus, a hybrid cloud offers flexibility.
- A public cloud is cost-effective; hence, a hybrid cloud is also cost-effective if the user wants to use the properties of the public cloud.

Disadvantages

- Complex networking problems may rise. Due to the complexity of having a public and a private cloud, there would be issues with configuring the network.
- The cloud must comply with the organisation's security rules. Both the public and the private cloud should comply with the organisation's security norms. It is not easy to set up clouds to meet this requirement.

4. Community Cloud

When different cloud services are integrated into a single cloud to meet the specific needs of an industry, community or business sector, the cloud is known as a community cloud.

The infrastructure of a community cloud is shared between organisations that have common concerns or interests. Industries such as healthcare and media opt for community clouds.

Here are some of the advantages and disadvantages of a community cloud.

Advantages

- The cost of maintenance can be shared among the organisations in the community.
- It is more secure than a public cloud and less expensive than a private cloud.

Disadvantages

- It is difficult to distribute the responsibilities among the organisations in a community.
- It is difficult to segregate the data among the organisations in a community.

Multi-Cloud Strategy

A multi-cloud strategy is when an organisation uses multiple public and private clouds from different providers. This is adopted in order to avoid lock-in with a single vendor and to make use of the best services offered by these cloud providers.

3. Cloud Service Models

On-demand cloud services are mainly of three types: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).

1. Infrastructure as a Service (IaaS)

These services are a set of computers, storage and a network, which are virtualised by cloud providers so that users can access and configure resources according to their needs. With IaaS, a user can rent IT infrastructure.

Common examples of IaaS include:

- AWS EC2,
- Google Compute Engine (GCE) and
- Digital Ocean.

Here are some of the advantages and disadvantages of IaaS.

Advantages

- The service provider provides the infrastructure, and the user has to just install an operating system of their choice and work on it.
- The user can modify the architecture as per their requirements since it is basic cloud infrastructure.
- The user has full control over all the computing resources.

Disadvantages

- There are security issues in an IaaS environment because of its multi-tenant environment.
- Outages at the vendors' end make it difficult for the users to access data for some time.

2. Platform as a Service (PaaS)

These are cloud services that provide an on-demand environment for developing and managing a software application. PaaS can be used to build, run and manage APIs (application programming interfaces).

Some common examples of PaaS include:

- Windows Azure,
- OpenShift and
- AWS Elastic Beanstalk.

Here are some of the advantages and disadvantages of PaaS.

Advantages

- **Prebuilt platform:** PaaS provides an already built platform for users to develop and run their applications.
- It is a simple model to use and deploy applications.
- **Low cost:** Since the platform is already built, the user needs to create only their applications. This reduces the costs related to hardware and software.

Disadvantages

- **Migration issues:** Migrating user applications from one PaaS vendor to another might raise some issues.
- **Platform restrictions:** The platforms provided by some vendors may have certain restrictions; for instance, the user can utilise only certain specified languages.

3. Software as a Service (SaaS)

These are cloud services that provide the users with a complete software application over the internet. All the infrastructure, application tools, data, etc. are located at data centres managed by the service providers.

Some of the examples of SaaS include:

- Google Apps,
- Salesforce and
- Dropbox.

There are two modes of SaaS:

- **Simple multi-tenancy model:** Each user has independent resources, which are different from the resources of other users.
- **Fine-grain multi-tenancy:** Resources are shared by several users, but the functionality of these resources remains the same.

Here are some of the advantages and disadvantages of SaaS

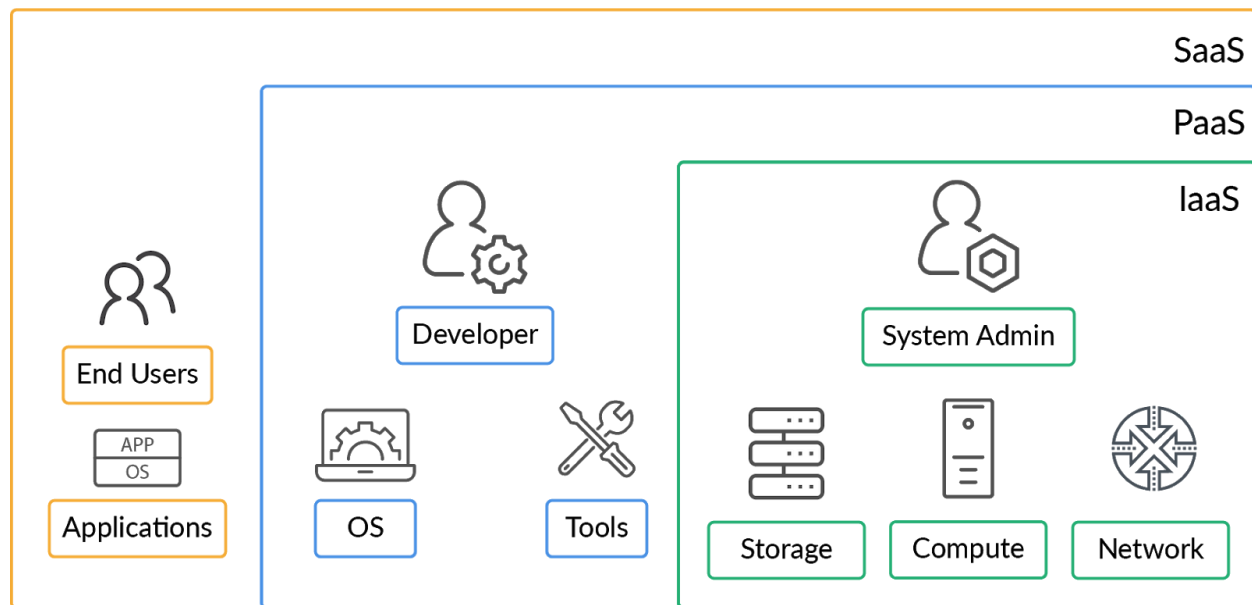
Advantages

- **Ease of access:** Users can access the applications on the server from anywhere using any internet-connected device. Most types of internet-connected devices can access SaaS applications.
- **Low maintenance:** Users need not update an application. The application is on the server and it is the service provider's responsibility to maintain the application.
- **Quick set-up:** Users do not need any hardware to install the application. The SaaS application is already present on the cloud.

Disadvantages

- **Lack of control:** Users do not have control over the SaaS applications. Only the vendor has full control of the SaaS applications.
- **Connectivity issue:** The applications can be accessed only via the internet. Thus, users cannot access the applications if there is no Internet.

This image illustrates the components of the different cloud service models.



4. Microservice Tools

In this segment, first, you learnt about various microservice tools, which are used to manage the microservices architecture. They include the following:

- **Message queuing:** Apache Kafka, AWS SQS
- **Containerization:** Docker
- **Orchestration:** Amazon Elastic Container Service, Kubernetes
- **Cloud deployment:** AWS, Google Cloud Platform, Azure
- **Application monitoring:** Prometheus
- **API testing and management:** Postman, API Fortress

Then you were provided with a brief overview of the different tools.

Message Queuing

- In a microservices service, we have autonomous services.
- To get the application running, they have to communicate with each other. For this, we have multiple tools available in the industry.
- For example, Apache Kafka, RabbitMQ, SNS
- Apache Kafka: It is a distributed Pub-Sub messaging system. It was originally developed at LinkedIn. It is agile, scalable and distributed by design.

- SNS: Simple Notification Service is an offering from AWS. It is a reliable, robust and flexible choice for microservice communications.

Containerization

- Containerization is the process of building an application along with all the required dependencies and configuration files together so that it can be run on any computing environment efficiently.
- Docker is an open platform for running and developing applications.
- Docker is the most common tool used in the Industry to build containers.

Orchestration

- It is the way to automate most of the manual work required to run containers in a production workload.
- Tasks such as scaling (up or down), networking, load balancing and many more are managed by container orchestration tools.
- Example: Kubernetes
 - It is the most common open-source container orchestration tool.
 - At present, it is the industry standard in terms of container orchestration.
 - It is built on more than 10 years of experience of running production workloads at Google, along with the best ideas and practices from the community.

Cloud Deployment

- Once your project is ready, you need to deploy it for use.
- The services available for this include AWS, the Google Cloud Platform and Azure
- AWS
 - Low-latency content delivery
 - Multiple global locations for deployment
 - Reliable, low-latency domain name resolution
 - Flexible pricing options
 - Extremely high scale
 - High availability

Application Monitoring

- Once we have your application ready, it is critical to see exactly what is going on in it at any given time.
- The tools available for this include Prometheus and Logstash.
- Prometheus: It is an open-source alerting and monitoring tool. It is mostly used to monitor your containers.
- Logstash: It is also an open-source tool, which helps to stash, centralise and transform data.

API Testing and Management

- Each individual service that we have in your microservices communicates with another using APIs.
- Hence, you must test each API and manage it to get the application running smoothly.
- The tools available for this include Postman and API Fortress:
 - Postman
 - It is a highly scalable API testing tool.
 - It allows the developers to easily test and run UI-driven API tests.
 - API Fortress
 - It is an API testing tool and a health tool, which helps to automate the process of health monitoring and load testing.

Segment 5: Demonstration on Installing WordPress on EC2

In this segment, you learnt how to install WordPress on an EC2 machine. First, you created an EC2 instance with Ubuntu AMI and then connected to the instance from your terminal; gave sudo access; updated the package; and then installed the Apache server, the MySQL server, PHP and the required PHP plug-ins. Then you created a database for WordPress, changed the document root, downloaded WordPress, modified the WordPress configuration and then tried to access it with your public address.

The step-by-step guide for this process is provided in this [document](#).