==========
Linux OS
=========

1) Linux OS

2) Linux Commands

3) What is IT Infrastructure

4) On-Prem Infrastructure

5) Cloud Computing

6) Advantages with Cloud

7) AWS Cloud

8) AWS Services Overview

9) EC2 (Elastic Compute Cloud)

- EBS
- Load Balancing
- Auto Scaling

10) S3 (Simple Storage Services)

11) RDS (Relational Databases)

12) IAM (User Management)

13) Bean Stack (Web application execution)

14) AWS Lambdas (Serverless Computing)

15) Billing Overview

==========
What is Operating System
==========

=> OS is a Mediator between user and computer

=> Wihtout OS we can't use any computer

=> Using OS we can run applications in computer.

Ex : Windows, Linux, Unix, Mac, Android, IOS etc.....

==========
Windows OS
==========

-> It is developed by Microsoft

-> It is GUI based OS (Graphical User interface)

-> It is commercial OS (we need to purchase)

-> Windows is single user based OS

-> Security Features are less (anti-virus is requied)

-> Windows is recommended for personal use

Ex: browsing, watch movies, play games, online classes.....

-> Windows not recommened to setup Infrastructure

Ex : Database, web server, app server, backup , storage etc...

==========
Linux OS
==========

-> It is community based OS

-> Linux is free & open source

-> Linux is Multi User based OS

-> Security features are very good in linux

(Anti-virus not required)

-> Linux is CLI based OS (Command Line Interface)

-> It is highly recommended to run servers

Ex : database, web server, docker, jenkins, k8s, sonar, nexus

===============
Linux History
===============

-> Linux OS developed by 'Linus Torvalds'

Unix OS

Minux

(Li) nus Torvals + Mi(nux) = Linux

-> Linus Torvalds released linux os into market with source code...

-> Many s/w companies downloaded linux source code and modified according to their requirement and released into market with their brand names those are called as 'Linux Distributions'

- Amazon Linux
- Ubuntu
- Red Hat (community & enterprise)
- Fedora
- CentOS
- SUSE
- Kali
- Debian ..... 200+

==================
Environment setup
==================

1) Create Account in AWS

2) Create Linux Virtual Machine using AWS EC2

3) Connect with Linux Virtual machine using MobaXterm/Putty

@@@ Linux VM Setup : https://www.youtube.com/watch?v=uI2iDk8iTps

==========
Linux CommandS
==========

whoami : To display logged in username

date : display current date

cal : display calendar with current month

pwd : display present working directory

ls : list files and directories of pwd

```
$ ls
$ ls -l
$ ls -lr
$ ls -lt
$ ls -ltr
```

cd : change directory

```
$ cd <dirname>

$ cd ..
```

mkdir : To create directory

```
$ mkdir java
```

rmdir : To delete empty directory

touch : To create empty files

```
$ touch f1.txt

$ touch f2.txt f3.txt f4.txt
```

cat : To create files + append data + display data

```
$ cat > java.txt

$ cat >> java.txt

$ cat java.txt

$ cat -n java.txt
```

cp : To copy file data

```
$ cp <src-file> <target-file>
```

mv : To rename files/directories + To move files/directories

```
$ mv f1.txt f11.txt

$ mv f1.txt mydata
```

```
rm : To remove files and non-empty directories

                        $ rm <file-name>

                        $ rm -r <dir-name>

head : Print top 10 lines of file data

                        $ head a1.txt
                        $ head -n 15 a1.txt

tail : Print last 10 lines of file data

                        $ tail a1.txt
                        $ tail -n 30 a1.txt

grep : Print file data based on pattern match

                        $ grep "AWS" a1.txt

                        $ grep -i "aws" a1.txt

Note: To see exception messages in log file we will use 'grep' command.

                        $ grep -i "exception" app.log

vi : Visual Editor (text editor)

                        $ vi <file-name>

                        command mode : only read + delete

                        insert mode : press 'i' (modifications allowed)

                        esc mode : press 'esc'

                                        :wq => write and quit

                                        :q! => close without saving

================================================================

ifconfig : To get ip address

ping : To check connectivity

wget : To download files using url

curl : To send HTTP request


==========================
Package Managers in Linux
==========================

=> Package managers are used to install softwares in Linux machine

                amazon linux / red hat : yum

                ubuntu / debian : apt

# install git client
$ sudo yum install git
$ git --version
```

```
# install maven
$ sudo yum install maven
$ mvn --version
$ java --version


=======================================================
How to deploy Spring Boot application in Linux VM
=======================================================

1) Clone Git Repo

                $ git clone https://github.com/ashokitschool/sb_log_app.git

2) Build app using maven

                $ cd sb_log_app
                $ mvn clean package

3) Run the jar file

                $ java -jar target/sb-log-app.jar

4) Enable Embedded Server port num in security group inbound rule

5) Access application in browser.

                        URL : http://public-ip:port/welcome

=========================================================================


=============
AWS Cloud
=============

1) What is Infrastructure

2) What is On-Prem infrastructure

3) Challenges with On-Prem Infrastructure

4) What is Cloud Computing

5) Benefits of Cloud Computing

6) AWS Introduction

7) AWS Services Tour

8) Working with AWS Services


The process of delivering IT resources over the internet with pay as you go model..


==============
AWS Services
==============

=> We have 200+ services in AWS cloud platform....

1) EC2 : Elastic Compute Cloud : To create virtual servers
```

2) S3 : Simple Storage Service : Unlimited Storage

3) RDS : Relational Database Service (Oracle, MYSQL)

4) Route 53 : Domain Name Service (domain mapping)

5) IAM : Identity & Access Mgmt (users, groups, permissions)

6) Beanstack : To run web applications (PaaS)

7) AWS Lambdas : Serverless computing (pay as you use)

==========
EC2 Service
==========

=> EC2 stands for Elastic Compute Cloud

=> EC2 is used to create virtual servers (machines)

=> EC2 is one of the most famous service in AWS cloud

=> EC2 is a paid service (hourly billing)

9:00 AM - 9:15 AM : 1 hour bill

9:30 AM - 9:40 AM : 1 hour bill

=> To encourage learners, AWS provided "t2.micro" as free of cost for 1 year. Monthly 750 hours.

================
EC2 Components
================

1) AMI : Represents softwares configuration (os, server...)

Ex : Windows AMI, Amazon Linux AMI, Ubuntu AMI....

2) Instance Type : Machine configuration

Ex: t2.micro(1GB), t2.medium(4GB), t2.large(8GB)....

3) Key Pair : Public key & private key

- AWS will store public key
- we will get private key

Note: one key pair we can use for multiple ec2 instances.

4) VPC : Network for EC2

5) Security Group : Firewall (inbound & outbound)

Windows => RDP => 3389
Linux => SSH => 22
Webserver => HTTP => 80
HTTPS => 443
MySQL => 3306

=> Inbound rules are used to allow incoming traffic
=> Outbound rules will allow outgoing traffic

Note : One security group we can use for multiple ec2 instances.

6) Storage : EBS Volumes (hard disc/ssd)

Windows : 30 GB (default)

Linux : 8 GB (default)

Note: Max capacity for EC2 VM is 16 TB.

==========
Types of IPs in AWS
==========

=> We have 3 types of IP's

1) private ip : Fixed IP address for internal communication.

2) public ip : Dynamic IP. For external communication.

13.127.226.25 (firtime)
35.154.28.3 (after restart)
15.207.24.103 (elastic ip)

3) elastic ip : Fixed public ip (it is commercial)

==========
Lab practice on Elastic IP
==========

Step-1 : Allocate Elastic IP (getting from AWS)

Step-2 : Associate elastic ip to EC2 instance

Step-3 : Re-start EC2 instance(ip will not change)

Step-4 : De-Associate elastic ip from ec2

Step-5 : Release elastic ip to aws.

==========
Load Balancer
==========

=> When we deploy our application in one server then below are the problem.

1) All request coming to one server

2) Burden will be increased on server

3) Response delay for customer

4) Server Crash due to heavy work load

5) Single Point of failure

Note: To overcome above problems we will use LOAD balancer in relatime to run our applications.

=> It is used to distribute load to multiple servers.

=> We have below advantages with Load Balancer...

1) App will run in multiple servers

2) Incoming requests will be distributed

3) Burden will be reduced

4) Responses will be faster for customers

5) High Availability of application

=> When request comes to Load Balancer, it will distribute the load to multiple servers in round robbin fashion.

==========
What is user data ?
==========

=> It is used to execute the script while launching EC2 VM.

Note: User data will execute only once when the machine is started.

```
#! /bin/bash
sudo su
yum install httpd -y
cd /var/www/html
echo "<html><h1>Life Insurance Server - 1</h1></html>" > index.html
service httpd start
```

==========
Process to Setup Load Balancer in AWS
==========

1) Create Security Group with below Protocols in Inbound Rules

SSH - 22
HTTP - 80

SGName : ashokit_Security_group

2) Create first EC2 instance (EC2-1) and Host Web Application

Note: Configure below script as 'User Data' at the time of launching the instance

```
#! /bin/bash

sudo su
yum install httpd -y
cd /var/www/html
echo "<html><h1>Life Insurance Server - 1</h1></html>" > index.html
service httpd start
```

3) Create second EC2 instance (EC2-2) and host web application

Note: Configure below script as 'User Data' at the time of launching the instance

```
#! /bin/bash
sudo su
yum install httpd -y
cd /var/www/html
echo "<html><h1>Life Insurance Server - 2</h1></html>" > index.html
service httpd start
```

4) Create Target Group and attach both ec2 instances created above

5) Create Load Balancer (ALB) and attach target group

Scheme : Internet facing

Select : Target Group

Listner : HTTP : 80

Security Group : ashokit_Security_Group

Note: Once Load Balancer created, DNS will be generated

5) Send a request to Load Balancer DNS URL and see the response

(it should distribute traffic to both servers)

==========
Note
==========
1) DELETE Load balancer
2) Delete EC2 instances

======================
What is Auto Scaling
======================

=> Auto Scaling groups are collections of Amazon EC2 instances that enable automatic scaling and fleet management features. These features help you maintain the health and availability of your applications.

=> We have below advantages with Auto Scaling

1) fault tolerance

2) cost management

3) high availability

======================
EC2 service Summary
======================

1) What is EC2 & why ?
2) What is AMI ?
3) What is instance type ?
4) What is Key pair ?
5) What is Security Group
6) What is EBS volume
7) How to launch and connect with Windows VM
8) How to launch and connect with Linux VM
9) Types of IPs in AWS
10) What is user data ?
11) What is Load Balancer
12) LBR setup with target group
13) Auto Scaling Group

==========
Spring boot with AWS RDS DB : https://www.youtube.com/watch?v=GSu1g9jvFhY
==========

==========
AWS RDS

==========

=> For every application database is required to store data permanently

=> We can use database in 2 ways

1) On-Prem database

2) Cloud Database

==========
Challenges with On Prem Database
==========

1) We need to take care of DB server

2) We need to download & install DB server

3) We need to purchase DB licenses

4) We need to secure database server

5) We need to maintain DB backup

Note: To overcome all theses challenges we will use Cloud Database.

==========
Cloud Database
==========

=> RDS stands for Relational Database Service

=> RDS is used create relational databases in the AWS cloud

Ex: Oracle, MySQL, Postgres, SQLServer.....

=> RDS is a fully managed service in AWS cloud

=> If we use RDS then AWS will take care of our Database Administration

Ex: License, security, backup ....

=> RDS works based on Pay as you go model.

##########
Steps to create MYSQL DB using AWS RDS
##########

1) Login into AWS management console

2) Goto RDS Service

3) Click on 'Create Database'

Choose a database creation method : Easy Create
Engine Option : MySQL
Template : Free Tier
DB instance Identifier : ihis (Note : you can give anything)
Username : admin
Password : Choose a passord

4) Click on 'Create Database' (It will take few minutes of time to create)

Note: Notedown username and password of the database

5) Once Database created, it will provide database Endpoint URL to access

6) Change Database to Public Access

7) Enable All Traffic in Security Group attached to Database.

==========
Database Credentials
==========

Endpoint : jrtp-rds-db.cx2weqx.ap-south-1.rds.amazonaws.com

Port : 3306

Username : admin

Password : AshokIT2024

Initial DB name : jrtpdb

==========
Lab Task
==========

Step-1 : Create MySQL DB using AWS RDS

Step-2 : Test DB Connection using Workbench

Step-3 : Configure RDS DB credentials in Spring Boot App & execute

Step-3 : Delete Database To avoid billing

========
AWS S3
========

=> S3 stands for simple storage service

=> S3 is used for unlimited storage

=> S3 works based on Object storage (object = file)

=> S3 is a paid service (for storing & for retriving)

=> S3 maintains data using buckets concept

=> Bucket means group of objects

ex : netflix_tollywood_movies_bucket
netflix_bollywood_movies_bucket
netflix_hollywood_movies_bucket

Note: When we create bucket, aws will generate one URL for that.

=> When we upload object in the bucket then object url will be generated.

=> Using S3 service we can host static website also.

==========
S3 Lab Practice
==========

1) Create bucket in s3

2) upload object in s3

==========
Static website hosting using s3
==========

=> Website means collection of web pages (html + css + js)

Step-1 : Create S3 bucket

- Enable ACL
- Allow Public Acc

Step-2 : Upload website files (index.html & error.html) as objects in bucket

- Grant Public Read Access

Step-3 : Enable Static Website hosting

(bucket -> properties -> Static website hosting)

Step-4 : Access website URL

==========
Assignment
==========

1) Develop spring boot web application to store course details

- course name
- course duration
- course price
- course image

Note: Course image should be stored into AWS s3 bucket and course info should be stored into db table.

Note: To implement this task we need IAM user with S3FullAccess & Security Credentials of IAM user.

Note: We need to use AWS SDK to perform this operation.

==========
AWS IAM
==========

=> IAM stands for identity and access management

=> It is used to manage users, groups and permissions

Note: Which user can access which service in AWS can be decided using IAM.

=> We can access AWS cloud platform in 2 ways

1) Root User (Super User)

2) IAM User (Limited Permissions)

Note: When we create account in AWs cloud by default we will get ROOT user account. Root user will have access for everything in AWS cloud.

=> It is not recommended to use Root account for daily activities.

=> For daily activities we will use IAM account.

=> In one root account, we can create multiple IAM accounts.

Note: In Real-Time, for every team member one IAM account will be provided with limited permissions.

==========
IAM Lab Task
==========

1) Login into AWS account as root user

2) Go to IAM service and create IAM user with below policies

1) EC2FullAccess
2) S3Fullaccces

Note: Provide web console access & generate security credentials

Note: Web Console access is used to login into aws account from browser.

=> Security Credentials are used to connect with AWS cloud using Program (Java/Python/DotNet)

==========

==========
Route 53
==========

=> It is a DNS service in AWS cloud.

=> DNS stands for Domain Name System

=> It is used to map our application with domain name.

=> DNS protocol works based on port number 53 thats why it is called as Route 53 service.

==========
Route 53 Lab Task
==========

@@@@ Domain Mapping :: https://youtu.be/f7bfbUPSONI?si=tqr4jlHeF6pQXW4Z

Step-0 : Host static website using s3

Step-1 : Check domain availability

Note: Based on extension price gets changed (least price is 5$)

Step-2 : Purchase Domain

Step-3 : Pay bill amount for domain registration

Note: With in 24 hours domain gets activated.

Step-4 : Map application URL to domain name.

Step-5 : Access our application using domain name.

==========
Elastic Beanstalk
==========

-> End-to-end web application management service in AWS cloud

-> In AWS, Elastic Beanstalk provides Platform As a Service (PaaS)

-> Easily we can run our web aplications on AWS cloud using Elastic Beanstalk service.

-> We just need to upload our project code to Elastic Beanstalk it will take care of deployment.

-> Elastic Beanstack will take care of softwares and servers which are required to run our application.

-> Elastic Beanstalk will take care of deployment, capacity provisioning, load balancer and auto scaling etc..

-> To deploy one java web application manually we need to perform below operations

1) Create Security Group

2) Create Network

3) Create Virtual Machine (s)

4) Install Java software in Virtual machine

5) Install Webserver to run java web application

6) Deploy application to server

7) Re-start the server

8) Create LBR

9) Create AutoScaling etc...

-> AWS providing infrastructure, we are creating platform using AWS infrastructure to run our java application (IaaS Model)

=> Instead of we are preparing platform to run our application, we can use Elastic Beanstalk service to run our web applications.

=> Elastic Beanstalk is providing Platform as a service.

==========
Advantages with Elastic Beanstalk
==========

1) Fast and simple to begin

2) Developer productivity

3) Impossible to outgrow

4) Complete resource control

==========
Create Sample Application
==========

-> Create Application

-> Choose the name

-> Select the platform

-> Create Elastic Beanstack with EC2 role

-> Create Instance Profile Role with Required Policies (AWSElasticBeanstalkWebTier)

==========
Below Beanstalk events
==========
Env Creation started
S3 bucket created to store the code
Security Group
VPC
EC2 instances
Webserver installation
Load Balancer
Autoscaling
Cloud watch

URL : http://webapp1-env.eba-rc8b64vg.ap-south-1.elasticbeanstalk.com/

==========
Elastic Beanstalk Pricing
==========

=> ThereÃ¢â‚¬â„¢s no additional charge for Elastic Beanstalk.

=> You pay for Amazon Web Services resources that we create to store and run your web application, like Amazon S3 buckets and Amazon EC2 instances, LBR and ASG.

==========
Procedure to deploy java-spring-boot-application
==========

1) Create one application in Elastic Beanstalk

2) Choose Platform as Java

3) Select Upload Your Code option and upload spring-boot-jar file

4) Select Required role

5) Select VPC and AZ's

6) Create Environment

-> Go to our application environment -> Configuration -> Edit Updates, monitoring, and logging -> Configure below Environment Property

SERVER_PORT = 5000

Note: After changing the port Environment will be re-started

URL : <Beanstack-domain-url>

==========
Application Versions
==========

-> In Beanstalk we can maintain mutliple versions of our application

```
                        sb-rest-api-v1.jar
                        sb-rest-api-v2.jar
                        sb-rest-api-v3.jar
```

-> We can deploy particular version of jar file based on demand.

==========
Serverless Computing
==========

-> Serverless computing means run the application without thinking about servers

-> AWS will take care of servers required to run our application

-> AWS lambdas are used to implement serverless computing

==========
AWS Lambdas
==========
AWS Lambda is a way to run code without creating, managing, or paying for servers.

You supply AWS with the code required to run your function, and you pay for the time AWS runs it, and nothing more.

Your code can access any other AWS service, or it can run on its own. While some rules about how long a function has to respond to a request, thereÃ¢â‚¬â„¢s almost no limit to what your Lambda can do.

The real power, though, comes from the scalability that Lambda offers you.

AWS will scale your code for you, depending on the number of requests it receives. Not having to build and pay for servers is nice. Not having to build and pay for them when your application suddenly goes viral can mean the difference between survival and virtual death.

==========
Running Java Code with AWS Lambda
==========

1) Create Lambda Function with 'java 11' runtime

2) Upload jar file in 'Code Source'

3) Configure Handler in Runtime

Class Name : in.ashokit.LambdaHandler

Method Name : handleRequest

Handler Syntax : className :: methodName

Ex: in.ashokit.LambdaHandler::handleRequest

==========
Linux + AWS Revision
==========

1) What is OS

2) Windows Vs Linux

3) Linux Distributions (amazon linux, ubuntu, red hat...)

4) Linux Virtual Machine

5) On-Prem Infrastructure

6) Cloud Computing

7) Cloud Providers

8) AWS Cloud

9) AWS Regions & Availability Zones

10) AWS Services Tour

11) EC2

12) Load Balancer

13) Auto Scaling

14) S3 (unlimited)

15) RDS

16) IAM

17) Elastic Beanstack (PaaS)

18) AWS Lambdas (serverless computing)

19) Route 53 (DNS)