DEMO:
Basic DevOps Tools & Linux Commands:

1. SCM (Source Code mangament)
   Tools : GitHub Git,, SVN, Bitbucket etc…
2. Build (Compile & Package)
   Tools : Ant, Maven, Gradle, Fastlane ….
3. CI/CD (Contineous integration & Conitneous Deployment & Delivery)
   Tools: Jenkins, Bamboo, Circleci, Teamcity etc…
4. CodeQuality (Validate Code syntax & Rules)
   Tools: Sonarqube
5. Artifactory (Storing Artifacts Backp)
   Tools : jfrog, Nexus
6. Deployment (Deployment & Configuration management)
   Tools: Ansible, Chef, Saltstack,Puppet,Appcenter etc…
7. Cloud
   Providers: AWS (Ec2, s3, sns, iam, LB,AS, VPC, Route53, CI/CD pipelines in aws etc,…)
   , GCP, Azure etc….
8. Docker (Containerization tool)
9. Kubernetes (Orchestraton tool)
10. Terraform (Infrastructure management tool)
11. Prometheus & Grafana & CLOUDWATCH (aws)
12. Linux Commands * basic Networking commands
13. XML, JSON, YAML, SHELL etc….

Day1:
Basic Linux Commands:

1. **pwd = present working directory (folder)**
2. **mkdir = make directory (create folder) (EX: mkdir demo)**
3. **touch = create the files (EX: touch filename)**
4. **cd = change directory (Ex: cd movies)**
5. **ls = list of files**
6. **ls -a = list of hidden files**
7. **ls -ll = long list of all files (ll = longlist)**
8. **cp source destination (EX: cp abcd.txt 123.txt)**
9. **mv source destination (By using this we can rename and move the files)**
   Ex: mv abcd.txt myfile.txt or. mv abcd.txt c/user/username/foldename
10. **rm filename (we can delete the file)**
11. **rm -rf foldername (we can delete files & Folder forcefully)**
12. **ipconfig = Windows (to find ip address)**
13. **ifconfig = linux & Mac**
14. **ipaddr = find the ipaddress**
15. **ping ipaddress (to check the server is up or down)**
16. **chmod (by using this we can give access permissions to the files & Folders)**
17. **grep = filter the process ids**
18. EX: ps -ef | grep java
19. **find = find the listed required files**

20. top = to see the running process occupied space
21. kill = to stop the process
22. ps -ef = present status of running applications
23. wget. : by using this we can download packages from internet.
24. Curl : by using this we can download & Upload packages to internet.
25. how to edit files and add the content in linux:
create a file = touch filename
vi filename
I = insert
Write content
Press esc
:wq! = save the data & quit.

25: . (dot) = present place or directory or folder
26. cd .. = to go one step back
27. cd ../../.. = to go three steps back
28. df -h = too see the diskspace
29. free -m = to see the memory free space

Day2:

SCM = Source code management

Repositories : (Storage)
Online repo or central repo or remote repo
Local repository

Pull or Clone = taking code from online repo to local repo (laptop)
Do the required changes in code and do the commit
Push to online repository

Tools : Git,Github, Bitbucket etc….

Gitbash (localy install). ----------→ Github (online repo)

Steps:
Create Github account in online?
Create repository in github?
Install git bash in local (laptop)?
Clone the code from github repository to local?
Do the required changes and add commit?
Push the latest code chabges to github?

Create Github account in online?
Go to google → use the below link → https://github.com/join?source=login
→ Username → Emailid → Password → Verify the account → create account → select student → learn to code → interested in DevOps → create account → verify email address →login into github account.
Create repository → repo name → description → public → initialize with readme file → create repository.

Download git bash for windows & install:

https://github.com/git-for-windows/git/releases/download/v2.30.1.windows.1/Git-2.30.1-64-bit.exe

1. install the git bash in your computer.
2. Go to (GITHUB) online repo copy the **CLONE** URL link.
3. go to desktop create folder.
4. go to folder give right click and open git-bash.

   # Configure Username & email
5. git config - -global user.email “emilid”
6. git config - -global user.name “name”

   #Clone the code from github repo
7. git clone repoURL-link
8. cd reponame
9. ls -ll
   #modify the files
   Create file = CMD: touch filename
   Edit and add the data in file = CMD: vi filename → I (insert) →write the data → press esc → :wq! (to save the data)
   Save the file

10. git status (red colour)
11. git add .
12. git status
13. git commit -m “message”
14. git push (if required give the username and password)

Note: after push go to your git hub account refresh the page and check the latest changes.

Realtime:
We are not push the code directly to main branch , we have to create the pull request by using any feature/bugfix/release branches , once PR is approved then we can merge the code form our branch to main branch.

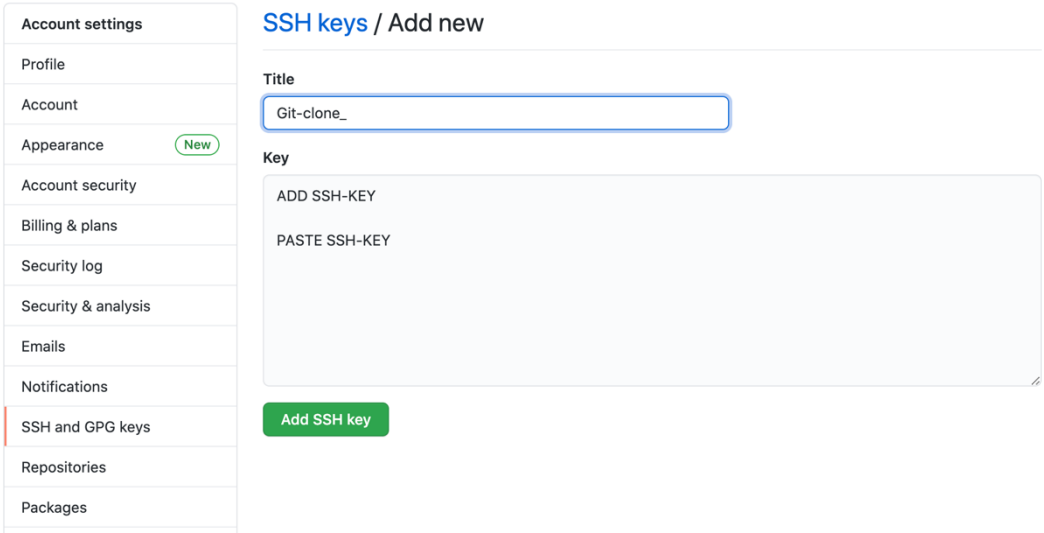
Branching strategy:
Develop branch
Bugfix branch
Release branch
Feature branch

Git Commands:
git config - -global user.email “emailid”
git config - -global user.name “username”
git clone https://github.com/devopsreddyprasad/DevOps.git
ls
cd DevOps/
ls

vi README.md
touch developercode
vi developercode
git status
git add .
git status
git commit -m "added data in readme file and create the new devcode"
git log
git push
history

SSH Connectivity with GitHUB:

1. by using this we can create password less authentication
2. To create the ssh key we use the below command
   CMD: ssh-keygen
   Note: CMD, GITBASH, Terminal etc…
3. SSH key will generate the PVT-KEY and Public-keys
   Note: Public key we will add in git-hub and other servers based on requirement.
4. SSH key defaukt location : /Users/Username/.SSH/*
5. How to add Publickey in github account
   Copy publickey → go to gitgub account → select user global settings → select SSH & GPG Keys → select SSH keys → Add New SSH-Key → Title = git_clone → in box we need add the Publickey → add SSH-key.



6. How to Clone By using SSH-Key:
   Select repository → select code → here select or click on SSH →Copy the SSH URL link → go to desktop on your local server → open git bash in required locations → Git clone ssh URL-LINk → give Enter → clone is done.

7. Perform your normal required git-hub operations and push the code to repository.
8. Go to Git-hub location and refresh the page and see the latest changes.

Build Tools:

Ant, Maven, Gradle, Fastlane.

#Maven advantages: & Difference B/W ant and maven?

1. We can create required files by using Maven command.
2. It have storage facility that means it will store all dependencies.
3. Plugins : it contains extra software functionalities to connect those servers.
4. Release the packages.
5. Ant don't have any project structure.
6. Ant and maven using the XML language.
7. Ant = build.xml, Maven = POM.XML & Settings.xml

1. Maven Lifecycle:
   Validate → clean → compile → test → package → deploy → install.
2. Maven Commands & Goals:
   CMD1: mvn validate
   CMD2: mvn clean
   CMD3: mvn compile
   CMD4: mvn test
   CMD5: mvn package
   CMD6: mvn deploy
   CMD7: mvn install
3. Maven Commands Execution:
   Calling = CMD6 (Deploy)
   Execution = validate + clean + Compile + test + package + deploy

*****5. Here we canot skip any commands in Maven, we can skip test cases only.

4. Maven Repos:
   1. Local repo or .M2
   2. Central repository
   3. Remote or OWN. Libraries

Maven repositories :

1. **Local repo(.M2): by using this maven will store all dependencies in local, once if we have dependencies then it will use form here, otherwise it will connect to central repository and download the dependencies and storing in .M2 repo.**
2. **Central repo: this is the repo is maintained by apache maven repositories.**
3. **Remote repo or (Artifactory): this repo is maintaining by our project team; there we will keep our project dependencies and securely.**

Maven Files:

1. **Setting.xml : by using this we can add plugins and configure (integrate) with other tools.**

2. **POM.xml : (project object model) by using this we can define our project details.**
   Ex: scm, G = groupid , A = artifact id, V= versions, Distributions, Dependncies, package details etc…..

   Installation of maven and working with maven:

   Step: maven have JDK dependency.

How to Configure Maven & JDK:

1. Oracle account to download the JDK file
   Link : https://profile.oracle.com/myprofile/account/create-account.jspx
   Note : add all required details and create the oracle account.
   JDK → JRE .
2. Downloaded & install JDK file.
   Link: https://www.oracle.com/in/java/technologies/javase/javase-jdk8-downloads.html#license-lightbox
3. Maven zip Download & Unzip .
   Link: https://mirrors.estointernet.in/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.zip

4. Copy Java & Maven path up to bin:
   Java : C:\Program Files\Java\jdk1.8.0_181\bin
   Maven : D:\DevOps_Softwares\apache-maven-3.6.3\bin

5. Configure Java & Maven in Environment variables:
   Go to this pc → Rightclick → Properties → Advanced System Settings → Environment Variables →
   Uservariables : (here create new and add java & Maven paths before bin
   EX: **JAVA_HOME** = C:\Program Files\Java\jdk1.8.0_181 ,
   MAVEN_HOME = D:\DevOps_Softwares\apache-maven-3.6.3
   → **System Variables** :
   JAVA_HOME= C:\Program Files\Java\jdk1.8.0_181\bin ,
   MAVEN_HOME= D:\DevOps_Softwares\apache-maven-3.6.3\bin.

6. Go to desktop and open cmd or gitbash.
7. Search for java & maven versions

   CMD: java –version , mvn - -version.

How to create POM.xml from Scratch:

1. Install and check the java & maven versions.
2. Go to c drive → Users → Username → open window and minimize.
3. Create folder (maven_operations)in desktop → go to folder →give right click →open gitbash →run the below connamds →CMD: "mvn archetype:generate" → choose version (example choose 3rd version) → give the group id (google.com) → artifactid

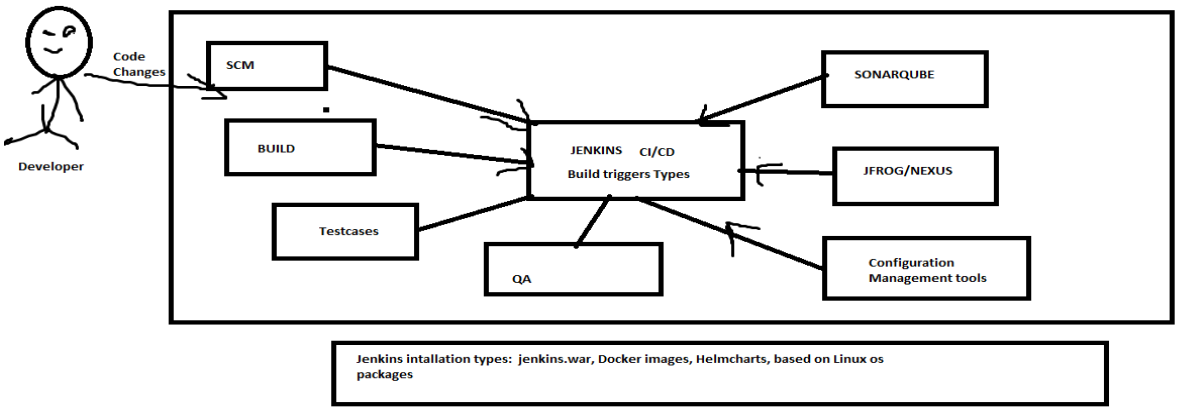
(com.google) → version (google) → packagetype = war → give "Y" for yes operation → build is success.

4. Go to maven desktop folder and go to pom.xml location → run the maven commands and observer the changes.

Commands:

153 ls
154 cd com.google
155 ls
156 mvn clean =. Sucess
157 mvn compile = success
158 mvn test = sucess
159 mvn package = success
160 mvn deploy = fail
161 mvn clean = here observe what it is cleaning
162 mvn package = here observer all stages of commands opeartions

CI/CD TOOLS:



CI = continuous integration
CD = continuous deployment or delivery

1. **Installation of Jenkins :**

Install Java → go to google → search Jenkins war download → Link : https://updates.jenkins-ci.org/download/war/ → Here choose version → Ex: 2.244 → War file download link : https://updates.jenkins-ci.org/download/war/2.244/jenkins.war → move this file into devops softwares folder → → open git-bash in Jenkins war location → Run the below command : java –jar Jenkins.war . here accept firewall settings →

OR

1. **Use the below link and download the Windows file https://www.jenkins.io/download/**

Once Jenkins is up and running → go to google and search for localhost:8080 → go to the below user location and copy the password (`C:\Users\Reddy prasad\.jenkins\secrets\initialAdminPassword`) → Continue → install

sugguested plugins → UN =admin → Password = admin → Confirm-password = admin → full name = admin -→ Email: sbrpdevops@gmail.com → Save & Continue → Save & Finish →Start using Jenkins.

Install plugins:
Create user account :
Configure tools:
Credentials:
Manage jenkins = 1. Configure systems
2.Global tool configurations
3. Manage Plugins
4.Manage credentials
5.Manage Users
6.Manage nodes
7.system properties etc…..

Job Types:
1. Freestyle job
2.Maven Job
3. Multi job
4. Pipeline job..

How to map jenkins portnumber: java -jar jenkins.war --httpPort=8081
How to install Plugins:

New-Day:

How to install Plugins: By installing this we can get new features

Go to Jenkins → Manage Jenkins → manage plugins → choose available → search for plugin → select the plugins → install without restart or install after restart.
Ex: git , maven integration, gradle, sonar, artifactory,docker, job config history etc…

Create Freestyle Job:
Go Jenkins → select New item → give job name → select freestyle job → ok.

Go to Job Configure:
Add description → add parameters → use git SCm to clone the code → add URL link → add credentials →add branch names → if required enable the cron tabs → Add Execute Shell & save job.

Build the job.

NewDay:

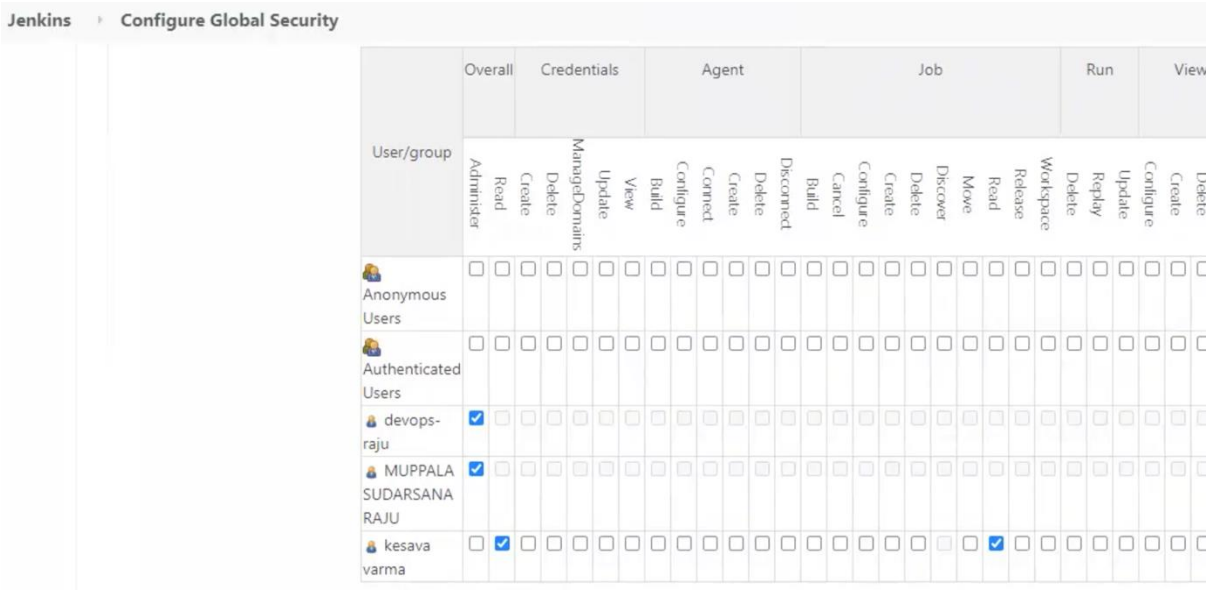
1. How to create Users and provide access ?

Jenkins ✍ manage Jenkins ✍ Manage Users ✍ create User ✍ Username, Email, Password, ConfirmPassword ✍ create User.

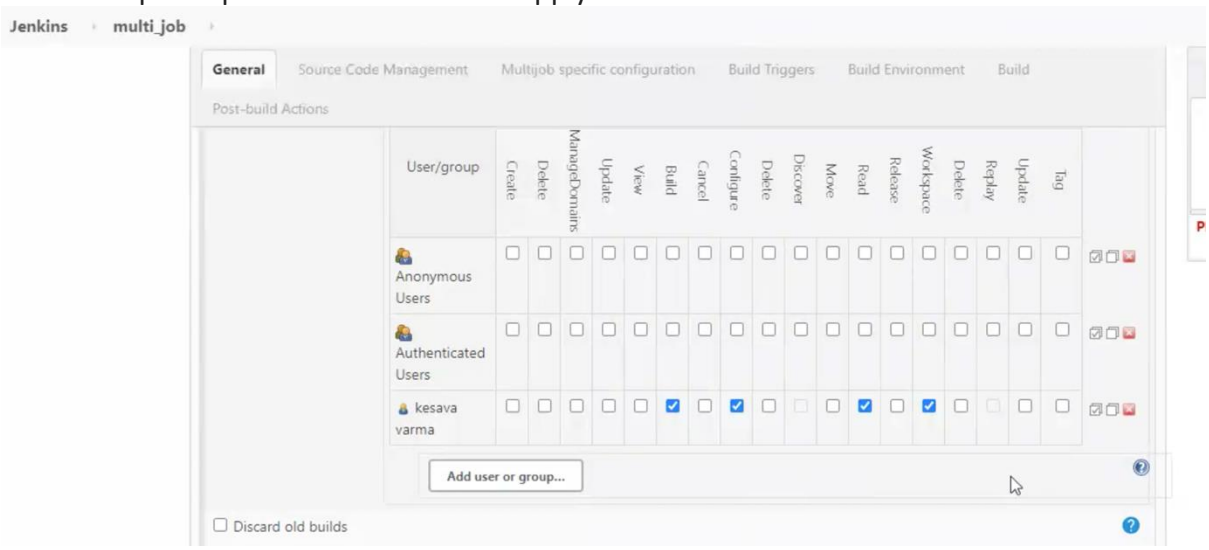
2. How to add Users in Global Security :

Jenkins ✍ manage Jenkins ✍Configure global tool security✍ Enable "Project-based authorization strategy" ✍ **Note: add your current admin user and provide administration access to admin-user, then add created user and provide admin access to them.**

✍ Create user and for this user we can provide limited access, for this see the below user "Kesava Varma" permissions. ✍ Save & Apply.



3. Go to Required Job✍ Select Configure ✍ Enable Project based security ✍ add user ✍ Give required permissions ✍ save & Apply.



#######################. ********** ###############

MASTER & SLAVE:

1. Enable the ssh configuration.
2. Create slave machine ?
3. Connect with slave and run a job in particular required slave machine?

Enable the ssh configuration:

Managejenkins → Configure global security →SSH SERVER → Select Fixed and port number = 50000 → Save & Apply.

Create slave:

Manage jenkins → Manage nodes & cloud → New Node → name = slave1 → select permanent agent → ok.

Name → Description: ---- → #no executors = 2 (2= 2 jobs at a time) → Remote Root Directory , path = Go to desktop → create folder = slave1 → go to inside the folder → copy the path → and paste the path in remote root directory location → Labels = give based on serer os = **windows or linux** → Usage → Launch Method = select Windows service → select all default → If required add any tools paths in environment variables → Save.

Note: Bring to slae machine is up if you able to do the required configurations in slave.

Run Job in required Slave:

Go to required job → Configuration → Select Restrict where this project can be run → select label expression → give required label or slave machine → save & Apply.

SONARQUBE :

Topics: SONAR:

1. **Projects**
2. **QualityProfiles**
3. **Rules**
4. **Qualitygate : q) blocker,major,critical, etc…**
5. **Administration , API,Users,Creating projects, install plugins etc…**
6. **Plugins : EX: jacocoa , Cobertura.**
7. **Integrate sonar with maven.**
8. **Sonar-project.propertiesfile**

SonarQube:

1. **Code quality Analysis 2. Quality profiles & Rules**

2. **Installation in windows, Linux, MacOS**
3. **Sonar-scanner**
4. **Login Admin/Admin**
5. **Projects**
6. **Quality gates & Quality profiles**
7. **Administration: Users Add/Delete, creating projects, add plugins, Api tokens, etc…**
8. **Sonar-Properties file.**

Download Version 6.6:
https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-6.6.zip

OR

Use this link for all required versions and download the required version.
https://binaries.sonarsource.com/Distribution/sonarqube/

Extract the file → /Sonar-Qube/sonarqube-7.4/bin/Select windows (windows-64/32) → Run sonar.bat file with administration → go to Browser → localhost:9000 → skip → login → UN = admin → PW= admin.

How to Provide User access:
Administration → Security → Users → Create User → UN →Full-Name → Email-id → Password → Create.
How to Modify User access: select user groups → enable permissions → Done.
How to reset user Password → User settings →change password.
How Create API-Token & Reset your Current User Password:
Go to User account → Security → Change Password.
Go to User account → Security → Give Api Token Key Name → Generate Api Token.

How to Create Quality-Gates:
By using this we can pass the Conditions based on issues, if the qulitygate is not passed the condition then Jenkins build will be break or stop.
Select QualityGates → create → qualitygate- name → create → add conditions.

***** How to create projects & Assign the Qualitygates and qulity profies & Check the Code Quality Errors: *****
How to create Project: Administration → Projects → Management → create new project → Project Name → Project Key → Create.
How to assign the Quality gates and profiles to project:
Select Project → Choose project administration → Select quality gate or Quality profile → Choose required quality gate or Profile → it will update automatically.

How to find project Issue & Types:

Choose required project → project wise issues → Seviarity → here we can see the Blocker, Critical, Major, Minor and Info issues with Developers name also.

JFROG (Artifactory) :

1. Jfrog is enterprise one, we have 30 days free trail.
2. It will support for multiple repositories systems
    a. Local repo
    b. Central repo
    c. Remote Repo
    d. Distribution repo
    e. Tools wise repository types we have
    EX: Maven, Gradle, Docker, Chef, Generic ……
3. Go to jfrog website and do the registration login, here we can get the License-Key email, for our 30 days free trail.
4. Download the required package and install the jfrog.
5. Jfrog default port number is 8081.
6. Here we can provide the access to other users also.
    Ex: Dev Team, DevOps team, QA team etc…….
7. Here we have to check the license key and validation timings also.
    JFROG (Artifactory):

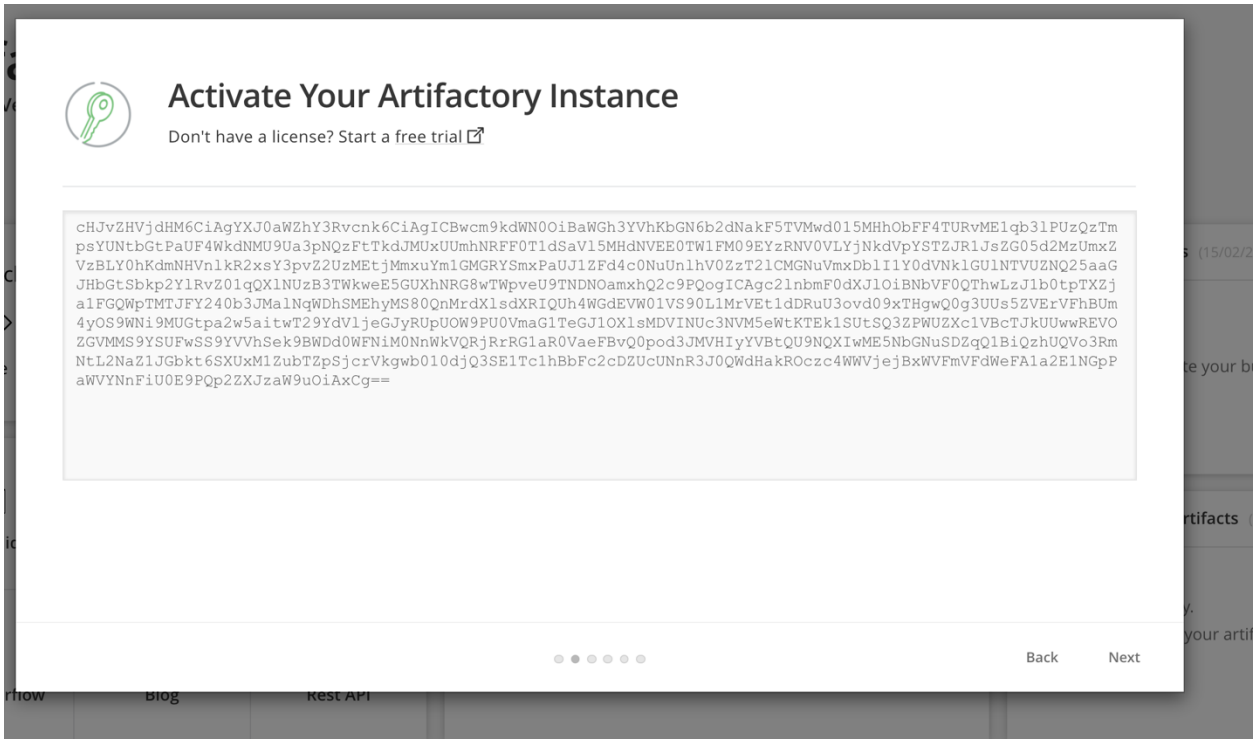
Download Links: 1. https://bintray.com/jfrog/artifactory-pro/jfrog-artifactory-pro-zip/6.21.02.

https://bintray.com/jfrog/artifactory-pro/jfrog-artifactory-pro-zip/6.23.7

How to get 30 days free- trail License-Key: https://jfrog.com/artifactory/start-free/

Installation:

1. Download artifactory → Unzip
2. Goto artifactory → upto bin → based on os run the file (if windows system give right click and run as administrator)
3. Go to browser → localhost:8081
4. Login UserName – admin, Password = password.
5. Click on next → Enter the license-Key → Next → Configure password → Skip →Finish..



What are the Types of Repositories We have and advantages?
Learn few details about repository Types.

How to create repo in Artifactory:
Goto Admin → Select Repo Type → +New → select package type → Generic → Repo Key → Public Description → Save & Finish.

How to upload files Manually & Download & Check the uploaded files details:

Artifacts → Select Repository → Deploy → Choose files → give the required directory path → Deploy.
Select artifacts → select repository → select actions → Native browser → here we can se the files uploaded details & We can download the files.

How to Add Users and provide permissions:

Select admin → security → Users → + New → UserName → Select Required Permissions → Password → create User.

AWS CLOUD:

AWS

Ec2-Launch: Linux :

Step1:

Go to services ✍ Ec2 ✍ launch instance ✍ select free-tier LINUX instance ✍ select instance type ✍ configure instance details ✍ add storage ✍ select security group ✍ add rule ✍ select all traffic ✍ here we can choose anywhere option ✍ add storage ✍ add tags ✍ download keypair ✍ launch ✍ view instance.

Note: after few minutes we can able to see our instance and give the required name.

Step2: How to connect instance.

By using git bash, putty and other third-party tools we can use to connect the Ec2 instance.

Select required instance ✍connect ✍select SSH client ✍ goto pem download location ✍ run the below command chmod 400 class instance.pem ✍ copy example and use the command to connect the instance ✍ now we are connected.

```
WKMIN7010066:Downloads redprasa$ chmod 400 mytest.pem
WKMIN7010066:Downloads redprasa$ ssh -i "mytest.pem" ec2-user@ec2-100-25-179-206.compute-1.amazonaws.com

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-55-214 ~]$ clear

[ec2-user@ip-172-31-55-214 ~]$
```

Go to root user and install below plackages:

sudo su
yum update -y
yum install java maven git -y
java -version
mvn --version
git -version
git --version
history

Topics:

1. Launch windows instance
2. Connect windows instance
3. Attach volume to windows instance
4. Create and attach volume.
5. Enable configuration in windows server to get the volume storage.

How to Configure JDK in Linux:

Step1: Install JDK:

1. yum --showduplicates list java-1.8.0-openjdk-devel
2. sudo yum install java-1.8.0-openjdk-devel.x86_64
3. java -version
4. export JAVA_HOME="/usr/lib/jvm/jdk-1.8.0-openjdk.x86_64"
5. PATH=$JAVA_HOME/bin:$PATH
6. source .bashrc
7. export PATH=$PATH:$JAVA_HOME/bin
8. echo $JAVA_HOME
9. echo $PATH

Note & Task: Learn advantages of Linux file system.

Linux File Syatem : bin boot dev etc home lib lib64 local media mnt opt proc root run sbin srv sys tmp usr var

Ec2-Launch: WINDOWS:

Step1:

Go to services → Ec2 → launch instance → select freetier Windows instance → select instance type → configure instance details → select security group → add rule → select all traffic → here we can choose anywhere or MYIP option → add storage → add tags → download keypair → launch → view instance.

Note: after few minutes we can able to see our instance and give the required name.

Step2: How to connect instance.

Select instance → Connect → Download RDP client → Get Password → Browse & Upload the Windows instance Keypair → Decrypt the password → Copy the Password → run or Install the downloaded RDP file → accept all firewalls →add password → continue → now we are able to see the Windows server launch.

How to attach volumes to Ec2 Servers:

1. Attach volume to Ec2 -Linix Server:

Step1:

Go to ec2 → select volume → create volume → volume name → storage capacity → select ec2 instance availability zone → if required snapshot give the name →create volume.

Step2: select volume → actions → attach → select ec2 instance → attach.

Note: we can use one volume at a time only one single ec2 server.

2. How to attach volume to windows instance:

Step1: Go to ec2 → select volume → create volume → volume name → storage capacity → select ec2 instance availability zone → if required snapshot give the name →create volume.
Step2: select volume → actions → attach → select ec2 instance → attach.
Note: we can use one volume at a time only one single ec2 server.
Step3: do the need full admin changes in windows server.
Go to windows server → open control panel → select catrgory = small icons →select administrative tools → computer management →Diskmanagement →select our volume give right click → select online → give again right click on our volume →now select the initialize the disk → select volume unallocated →give right click → new simple volume → next → next → next →next → finish.

S3 Buckets :
Services ✍ s3 buckets ✍ create bucket ✍ bucketname ✍ selct region ✍ if required choose existing bucket settings (optional) ✍ block all public access ✍versioning enable ✍ create bucket.
Select s3 bucket:
Upload objects ✍ upload ✍ add file or folder ✍ choose file or folder ✍ upload.
What is s3 bucket lifecycle advantages?
What is s3 glacier and advantages ?
Try to lear s3 plugin configuration with Jenkins ?
Difference Between s3 bucket and AWS-CODE Artifact ?
Reference Link: https://www.serverkaka.com/2018/08/upload-build-aws-s3-from-jenkins.html

1. IAM (Identity and Access Management):

Learn about USERS, ROLES, GROUPD and Policies advantages.
Learn about MFA and download Microsoft mfa authenticator app.
Assign MFA login to your AWS account.

How to create Groups:
Go to services → search for 'IAM' → Choose the Groups → create group → group name → next step → attach policies → select the policies → nextstep → create group.
Select group → actions → edit group name or add or remove users → select users → add or remove User.

How to create USERS:
Go to IAM → select Users → create user → Username → select the access type 1. Programmatic access 2. AWS management console access → Console password → Next permissions → select gropus to add or create new group and provide the access or → tag → create user → **Note**: share the below details to users : 1. copy the URL link and share with them → username → password → accesskey and secret key.

How to create Roles:

By creating this roles we will attach to AWS services,
Ex: once attached the role to ec2 then ec2 server will perform those operations based on role access.

Services → IAM → select roles → create role → choose service type → next step → permissions → select policies → tag → Review → Role-name → create role.

How to attach role to Ec2:

Select Ec2 instance → actions → instance settings → attach or replace IAM Role →select I am role → attach or proceed.

Policies:

Policies are the permissions to provide in limits. We have different policies which is provided by the AWS.
If we need we can create our own policies also.

Autoscaling:

Step1:
Using below template we can define required ec2 servers with default updated configurations.

Create launch template:

Goto Ec2 → select autoscal groups → create autoscaling group → given autoscale name → choose create launch template → add the required name → add the required version → add template tags if required → select Application and OS Images → here browse and select the required os → select instalnce type → add and download the required keypair → in network add or select the subnet → add the security group & rules → keep default configurations → create launch template. → done.
$$$$$$$$$$NOTE : to find your template is working or not, please do the below testings $$$$$$$
Select template → actions → launch instance from template → launch instance. → go to ec2 dashboard check the instance is launching or not.

Step2:

Create Autoscaling group:

Go back to AWS Autoscaling groups → create Autoscaling group → Name → select our launch configuration template → Next → if required select instance purchase options → network → required subnets selection → Next → Next → add the required instance details desired capacity = 3 , minimum capacity = 3 , Maximum capacity =3 → next → Add Notifications → SNS topic → create new topic → add name & Email id → next → tag → next → verify all details and edit if required (or) create autoscaling groups.

Go to autoscale group check the details → go to EC2 Dashboard → check the servers are launching or not.

######### FOR MORE DETAILS#####

Create Launch Configurations :

Go to Ec2 → Select Autoscaling groups → Create Autoscaling Group → Name → choose the required launch configurations → **note:** if you don't have exist templates create new template → create Launch new template → Launch Template Name → Launch Template version description → Select AMI image → Select Instance Type → Create Keypair → Select Network Security group after launching the autoscaling groups or modify the existing security group and add inbound and outbound

rules → storage → add volumes → tags → **Note:** if required use the advanced variables → create Launch Configuration → View Launch Template.

##

Load Balancer:

Step1: create the required servers (3 +-..)
Step2: connect to the servers and check the security groups.
Step3:

1 yum install java httpd -y
2 use this commands to start the httpd service:
3 sudo service httpd start
4 sudo service httpd status
2 vi /var/www/html/index.html
Add the required data in the file & save.
3 service httpd status
4 service httpd start
5 history

add the webpage content data in index.html
###EX: go to browser, select ctrl+u , copy the content and edit and replace (google = your reqired name→ add this content to index.html and start the service → copy instance ip address → search in google browaer → here we can see our data.)

Step4: go to instances copy public DNS name and go to browser check the webpage data.

Step5: go to loadbalancer → select classic load balancer → create load balancer → loadbalancer name →Assign security groups → select security groups → Configure security settings → select check the health → add instances → add the tags → Review and Create → create → select load balancer → after 3 mns select load balncer → check the instances → instances must be in available or active state → copy the loadbalncer DNS name →go to browser paste the link → and do refresh to get the all server web data.

VPC : Virtual Provate Cloud

What is IGW, DNS, ROUTETABLE, NACL , SUBNETS, CIDR ?

How to create VPC and launch Ec2 server in VPC:

Select services → VPC → Create VPC → VPC Name → Define CIDR BlockRange → Ex CIDR = 12.0.0.0/16 → create VPC.

Create Subnets :
Based on requirement we can select and define the subnets.
Create subnet → choose VPC → subnet name → CIDR range of subnet → create subnet.

RouteTable :

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

Select route table → routetable name → select vpc → create route table.

IGW:

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

Select IGW → create IGW → Name → creae IGW.
Select the IGW → actions → attach to Vpc → Select VPC → attach InternetGateway.

Launch Ec2 in VPC:

Select Ec2 → launch instance → select instance Which Required → select instance Type → Configure instance details → tag → storage → security groups → Keypair → launch → View instance .

Select and find the Private IP address range , If it is under our VPC CIDR range then this VPC is working fine.

Note : if you are not able to see the Public DNS host details, do the below changes.

Go to VPC → select VPC → actions → Edit DNS host Name → Enable → save or apply the changes.

AWS ROUTE53:

Go to services → search route53 → select required services (Ex: DNS management, Hosted-Zones, Traffic Management etc…) → select DNS management → create hosted Zone → Domain Name → Discription → select type (Pub or Pvt) → Add Tags → create → DNS is created.

DNS NAME REGISTRATION:

Select DNS registration →give the required DNS name & select the domain based on cost or requirement → select DNS registration names → add to cart → continue → check bill and proceed to register the name & need to pay the money also → after payment will get the DNS name use that and attach to required applications.

AWS CI/CD Pipelines:

Code-commit:

Go to services → search codecommit → select codecommit → create repository → repo name –> repo description → if required enable codeguru → create → repo is done.
Select and perform the required operatons using SSH or HTTPS

Code-build:

Go to services → search codebuild→ select codebuild → create build project → name → description → enable the build badge → Restrict number of concurrent builds this project can start → add required count (ex: 3builds) → add tags →select source → select your scm code tool → add repository name → select branch or tag or commit → select environment → select the operating system based on server → runtime is standard & select the image-name →select required environment type → if required select artifactory → select monitoring → create build project.

Code-deploy:

Go to services → select code deploy → create application → application name →select type of compute instances (EC2, Lamda and ECS) →create application.
Create deployment configurations → deployment configuration name → compute type instances (EC2, Lamda and ECS) → select healthy instances → default 1 (we can give percentage or number upto 99) → create configuration deployment.
If required please create the notifications also → select notification rules → name → full details or basic detals → slect always deployment success, failed or etc…. →create SNS topics → create notification rules.

Code-pipeline:

Select code pipelines → create pipeline → name →create or select required roles → select advanced settings → select deployment options & Encryption options → next →select source provider → ex: aws code commit or other scm tools → select repo & branch → nect → create build project or choose existing one → next →select the deploy stage → select required type Ex: S3 → choose bucket name & path → next → create pipeline →now see the all stages functionalities → Run the CI/CD Pipeline Jobs.

Codeartifact:

Go to services → search codeartifact → create repository → name → description →Public upstream repositories (maven etc..) → next → select this AWS account → give domain-name → next → create repository.
Go ot repositories → select repository → view connection instructions and select the maven → copy and update the configurations in Maven Pom.xml or Settings.xml.
USE THE MAVEN COMMANDS TO BUILD & DEPLOY

KUBERNETES:

1. installation purpose use the below link
2. https://phoenixnap.com/kb/install-minikube-on-centos

USE THE SHARED EKS DOCUMENT TO CREATE KUBERNETES CLUSTER IN AWS.
Kubernetes Basic Commands:

Note: To crate the deployment purpose please use the below commands and before this create the required YAML files (EX: deployment.yaml, Service.yaml and pod.yaml etc…..)

```
# to see the all details like deployments, services etc,….
640  kubectl get all

# to see the list of pods
  641  kubectl get pods

# to see the list of services
  642  kubectl get svc

#to see the list of deployments
  644  kubectl get deployments

# to see the list of namespaces
  645  kubectl get namespaces
# create the deployment yaml files
  649  vi deployments-definition.yml
```

######*********###############

Yaml-Code :

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: testapp-deployment
  labels:
    app: mywebsite
    tier: testapp-deployment
spec:
  replicas: 5
  template:
    metadata:
      name: myapp-pod
      labels:
        app: myapp
    spec:
      containers:
        - name: nginx
          image: nginx
  selector:
    matchLabels:
      app: myapp
```

#########*********############

create the service.yml file to access deployment applications
651 vi service-definition.yml
##########**********##########
Yaml-Code:

```
apiVersion: v1
kind: Service
metadata:
  name: devops3-deployment
  labels:
    app: myapp
spec:
  type: NodePort
  ports:
    - port: 80
      targetPort: 80
      nodePort: 30004
  selector:
    app: myapp
```

##########*********########

to create the deployments use the below command
654 kubectl apply -f deployments-definition.yml
Note: deployments-definition.yml =File name of yaml

to check the deployment is created or not
655 kubectl get deployments

to get the pod & Service details
656 kubectl get pod
657 kubectl get svc

to access the application use the below link to get the URL link
658 minikube service testapp-deployment --url

######## goto Browser and Paste the URL link ########

################### KUBERNETES COMMANDS ####################
[ec2-user@ip-172-31-50-126 .kube]$ history
247 sudo yum update -y
248 sudo yum -y install epel-release
249 sudo amazon-linux-extras install epel
250 sudo yum -y install libvirt qemu-kvm virt-install virt-top libguestfs-tools bridge-utils
251 sudo systemctl start libvirtd
252 sudo systemctl enable libvirtd
253 systemctl status libvirtd

```
254  whoami
255  sudo usermod -a -G libvirt $(whoami)
256  sudo vi /etc/libvirt/libvirtd.conf
257  sudo systemctl restart libvirtd.service
258  wget https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
259  ls
260  chmod +x minikube-linux-amd64
261  sudo mv minikube-linux-amd64 /usr/local/bin/minikube
262  ls
263  minikube version
264  curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s
https://storage.googleapis.com/kubernetes-
release/release/stable.txt`/bin/linux/amd64/kubectl
265  ls
266  chmod +x kubectl
267  sudo mv kubectl  /usr/local/bin/
268  kubectl
269  kubectl version --client -o json
270  minikubestart
271  minikube start
272  docker --version
273  sudo service docker status
274  sudo service docker start
275  sudo service docker status
276  docker images
277  sudo chmod 777 /var/run/docker.sock
278  docker images
279  minikube start
280  kubectl cluster-info
281  minikube dashboard
282  kubectl get all
283  kubectl get no
284  kubectl get pods
285  kubectl get ns
286  kubectl get rs
287  kubectl api-resources
288  kubectl get all -o wide
289  kubectl get svc
290  kubectl get deployments
291  history
 39  kubectl get services
 40  kubectl get pods
 41  kubectl get ns
 42  kubectl create ns dev_env
 43  kubectl create ns devenv
 44  kubectl get ns
 45  kubectl get nodes
 49  kubectl top node minikube
 51  kubectl --help
```

```
52  kubectl get api-resources
53  kubectl api-resources
55  kubectl get all
56  vi deployment-nginx.yaml
57  vi service-nginx.yaml
58  kubectl create -f deployment-nginx.yaml
59  kubectl get deployments
60  kubectl get pods
61  kubectl logs testapp-deployment-b478cc546-28tr7
63  kubectl describe pod testapp-deployment-b478cc546-28tr7
64  kubectl get svc
65  kubectl create -f service-nginx.yaml
73  kubectl get svc
74  minikube service devops3-deployment --url
    Note: devops3-deployment = service Name
75  ping http://192.168.49.2:30008
76  kubectl get svc
82  minikube dashboard
83  kubectl get pods
84  kubectl get svc
86  minikube service s --url
87  kubectl delete svc devops3-deployment
88  kubectl get svc
####### TO SEE THE CLUSTER AUTHORIZATION DETAILS #########
89  cd ~
90  ls -a
91  cd .kube/
92  ls
93  vi config
100  cat /home/ec2-user/.minikube/ca.crt
```

######### USE THIS TO SEE ALL K8S API-RESOURCES #####

kubectl api-resources

PRACTICE TASKS

1. What is SDLC & TYPES & ADVANTAGES.
2. Create central repo and keep all project data in repo.
3. Clone and push to central repo and raise MR or PR then merge the changes.
4. List out the build tools and supporting languages
5. Create maven pom.xml and create own package.
6. Install and configure sonar & jfrog.
7. Install Jenkins & configure using Jenkins CI/CD pipeline with the below tools (git, maven, shell, sonar, artifactory etc..)

8. Create ec2 servers and deploy demo web application and take backup of EBS and AMI ?
9. Create VPC and Deploy EC2 server in VPC , and run web application application in public servers
10. Create VPC and Deploy EC2 server in VPC , and run Jenkins server
11. Create Loadbalancer and autoscale for using httpd webapplication
12. Create AWS CI/CD Pipeline from Maven builds & deploy file to Code-artifact
13. Install terraform & provisioning with AWS & launch the EC2, S3, IAM, VPC, EKS cluster.
14. Write docker file to deploy our own jar package to build image.
15. Create docker network and run the application containers in docker networking.
16.