**­­­DEMO:**

**Basic DevOps Tools & Linux Commands:**

1. SCM (Source Code mangament)

Tools : GitHub Git,, SVN, Bitbucket etc…

1. Build (Compile & Package)

Tools : Ant, Maven, Gradle, Fastlane ….

1. CI/CD (Contineous integration & Conitneous Deployment & Delivery)

Tools: Jenkins, Bamboo, Circleci, Teamcity etc…

1. CodeQuality (Validate Code syntax & Rules)

Tools: Sonarqube

1. Artifactory (Storing Artifacts Backp)

Tools : jfrog, Nexus

1. Deployment (Deployment & Configuration management)

Tools: Ansible, Chef, Saltstack,Puppet,Appcenter etc…

1. Cloud

Providers: AWS (Ec2, s3, sns, iam, LB,AS, VPC, Route53, CI/CD pipelines in aws etc,… )

, GCP, Azure etc….

1. Docker (Containerization tool)
2. Kubernetes (Orchestraton tool)
3. Terraform (Infrastructure management tool)
4. Prometheus & Grafana & CLOUDWATCH (aws) OR DataDog
5. Linux Commands \* basic Networking commands
6. 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**

1. **rm filename (we can delete the file)**
2. **rm -rf foldername (we can delete files & Folder forcefully)**
3. **ipconfig = Windows (to find ip address)**
4. **ifconfig = linux & Mac**
5. **ipaddr = find the ipaddress**
6. **ping ipaddress (to check the server is up or down)**
7. **chmod (by using this we can give access permissions to the files & Folders)**
8. **grep = filter the process ids**
9. EX: ps -ef | grep java
10. **find = find the listed required files**
11. **top = to see the running process occupied space**
12. **kill = to stop the process**
13. **ps -ef = present status of running applications**
14. **wget. :** **by using this we can download packages from internet.**
15. **Curl : by using this we can download & Upload packages to internet.**
16. **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**

1. **cd .. = to go one step back**
2. **cd ../../.. = to go three steps back**
3. **df -h = too see the diskspace**
4. **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

1. git config - -global user.email “emilid”
2. git config - -global user.name “name”

#Clone the code from github repo

1. git clone repoURL-link
2. cd reponame
3. 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

1. git status (red colour)
2. git add .
3. git status
4. git commit -m “message”
5. 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…

1. 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.

1. SSH key defaukt location : /Users/Username/.SSH/\*
2. 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.

Graphical user interface, text, application, email

Description automatically generated

1. 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.

1. Perform your normal required git-hub operations and push the code to repository.
2. 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
8. Maven Lifecycle:

**Validate 🡪 clean 🡪 compile 🡪 test 🡪 package 🡪 deploy 🡪 install.**

1. Maven Commands & Goals:

CMD1: mvn validate

CMD2: mvn clean

CMD3: mvn compile

CMD4: mvn test

CMD5: mvn package

CMD6: mvn deploy

CMD7: mvn install

1. 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.

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

1. Downloaded & install JDK file.

Link: [https://www.oracle.com/in/java/technologies/javase/javase-jdk8-downloads.html#license-lightbox](https://www.oracle.com/in/java/technologies/javase/javase-jdk8-downloads.html" \l "license-lightbox)

1. Maven zip Download & Unzip .

Link: <https://mirrors.estointernet.in/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.zip>

1. 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

1. 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.

1. Go to desktop and open cmd or gitbash.
2. 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