Important Steps of DevOps Lab

Exp2: -Installation of Git Bash

Steps: -

1. Install Git Bash
2. Create Account on GitHub
3. Create a Repository
4. Commit Changes in the Repository

Exp 3: - Git Bash Commands, Push, Pull, Fork

Steps: -

Open Git Bash

Commands: -

$ git --version
$ git init (Use in a specific folder)
$ git status
$ git config --global --list
$ git config --global user.name 'Demo'
$ git config --global user.email 'Demo@gmail.com'
$ git config --global --list
$ git add Demo.txt (Use after creating text file)
$ git commit -m 'first commit'
$ git status
$ git log
$ git log --oneline

For Push Pull Operation: -

$ git remote add origin https://github.com/Vinit7796/GIT.git
$ git remote show origin

Push Operation: -

$ssh-keygen -t rsa -b 4094 -C “salvivinit49@gmail.com” (For key Generation)

$ git add gitkey
$ git add gitkey.pub
$ git push -u origin master

Pull Operation: -

$ git pull origin main

Branch Merging : -

1. Create a Branch in Repository
2. Create a file ….After that Commit the changes
3. Select Compare & Pull option
4. Create Pull Request
5. Merge Pull Request

Fork: -

Use https://github.com/TheCurryMan/GithubIntro and fork.

Exp 4: - Jenkins Installation and Sample Project

Steps: -

1. Install Jenkins
2. Install/Download Maven
3. Edit Environment Variables
4. Open Jenkins and Install “Maven Integration” Plugin
5. Config Jdk,Maven,Git and credentials
6. Create Freestyle Project
7. Provide Repository URL and Credentials & change /***master** branch (if required)
8. Build Project

Exp 5: - Role Based Authorization

Steps: -

1. Create a User
2. Install Plugin ‘Role-based Authorization’
3. Manage Jenkins->Security->Authorization(Select role based)
4. Manage and Assign Roles->Manage Role(Add a Role e.g ”Programmer”)

5. Manage and Assign Roles->Assign Role(Assign the Programmer Role to the Created user)
6. Give different access to the user using ‘Manage Role’
7. Sign in using created User

Exp 6: - Pipeline

Steps: -

1. Create a Maven Archetype Project (Select first Archetype)
2. Create java directory in src->main directory
3. Create a java class in java directory (Enter code)
4. Run java class file
5. Enable Version Control
6. Commit Changes
7. Share project on GitHub (public)
8. Create Maven Project in Jenkins
9. Set Repository path & Credentials
10. Build Project

Exp 9: - Trigger Remotely

Steps: -

1. Install Plugin “Build Authorization Token Root” in Jenkins
2. Create Free Style Project in Jenkins
3. Configuration->Build triggers->Trigger builds remotely->mytoken
4. Use this URL: -
5. buildByToken/build?job=(Project Name)&token=mytoken
6. Paste this URL after localhost:8080/
7. It will build the project remotely.