DL Prac manual

Exp2 (To perform installation of Git and work on local and remote Git . repositories.)

1. **sudo su root**
2. **sudo apt update**
3. **apt install git**
4. **git –version**
5. **git config - -global user.name “omkar11”**
6. **git config - -global user.email “omkar11@gmail.com”**
7. **git config - -list**
8. To Integrate Git account with Github:
   Goto www.github.com > Sign-in to your account >Create a Repository(Public)
9. Now Clone the Remote Repository into your Local Repository
   - **mkdir workspace**
   - **cd workspace**
   - **git init**
   - **ls**
   - **git clone <link of your git repository>**
     - *Ex: - git clone https://github.com/omkar11/hello_test.git*
   - **ls**
   - **cd hello_test**
   - **touch hello.py**
   - **gedit hello.py:**

```
def factorial(n):
    if n < 0:
        return 0
    elif n == 0 or n == 1:
        return 1
    else:
        fact = 1
        while(n > 1):
            fact *= n
            n -= 1
        return fact
num = 5
print("Factorial of",num,"is",
factorial(num))
```

10. git commands to add, commit and push hello.py file to remote repository:
    - **git status**
    - **git add hello.py**
    - **git commit -m “First Python File”**
    - **git push origin main**
11. If authentication failed:
    - Goto your Github Account→User Profile→Settings→Developer Settings→Personal Access Token→Token(classic)→ Generate New
    - Click on Generate Token
      Copy the token and paste it some location:
    - **git remote set-url origin https://tokenhere@github.com/user_name/repo_name**
    - Again try to Push your changes to Github Repository
12. Go to github and verify if repository has been updated and hell.py has been added

Exp3 (To understand and perform version control system / source code . . management using Git)

1. Perform all steps of exp2 to clone a git repo and enter inside it by **cd hello_test**
2. **git branch feature**
3. **git checkout feature**
4. **git branch**
5. **echo “text from feature branch” >> hello.py**
6. **cat hello.py**
7. **git status**
8. **git add hello.py**
9. **git status**
10. **git commit -m “Added a line in feature branch”**
11. **git log**
12. **git checkout main**
13. **git branch**
14. **git merge feature**
15. **cat hello.py**
16. If git is not connected to github,create a classic token
    git remote set-url origin https://tokenhere@github.com/user_name/repo_name
17. **git push -u origin main**
18. Verify changes on github

19. **git branch -d feature**
20. **git branch**

Exp4 (To install and configure Jenkins to test and deploy an application with … Maven.)

1. **If Jenkins is not Installed**
    - **sudo su root**
    - **apt-get update**
    - **apt-get install openjdk-11-jdk**
    - **java - -version**
    - **curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo tee **
        - **/usr/share/keyrings/jenkins-keyring.asc > /dev/null**
    - **echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] **
        - **https://pkg.jenkins.io/debian-stable binary/ | sudo tee **
        - **/etc/apt/sources.list.d/jenkins.list > /dev/null**
    - **sudo apt-get update**
    - **sudo apt-get install jenkins**
    - **jenkins - -version**
    - **sudo systemctl start jenkins.servce**
    - **sudo systemctl status jenkins**
    - **sudo ufw status**
    - **sudo ufw allow 8080**
    - **sudo ufw enable**
    - Get password using **sudo cat /var/lib/jenkins/secrets/initialAdminPassword**
2. Log in to your jenkins account from browser (**localhost://8080**)
3. Go to Manage Jenkins > System > Change system Message to your name
4. Click on New Item > Freestyle Project > write description
5. On the source code management choose **Git** option and specify the repository URL in which you have added a java code
6. In Build Steps:
    - **javac Simple.java**
    - **java simple**
7. Apply and save
8. Click on Build now
9. <u>If any error in running java file</u>
    - In Source code management choose None
    - In Build Triggers > Build Periodically > schedule > type *****
    -

- In Build steps : **mkdir happy**
  cd happy
  touch cool.txt

10. In terminal ,
    - **cd /var/lib/jenkins/workspace/execute**
    - **ls**
    - **cd happy**
    - **ls**

Exp5 (To create and build a CI/CD pipeline in Jenkins to test and deploy an . . application over the tomcat server)

1. **sudo su root**
2. **find / -type f -name java**
3. *Skip below steps 4- 7 if they are already configured*
4. Copy the location path of openjdk and go to jenkins
5. Manage Jenkins >Tools>Name: **java_home** >JAVA_HOME: **path that you copied**
6. Git > path to git executable : **git** > tick install automatically
7. Maven > Name: **maven_home >** Install automatically > version : **3.8.3**
8. Dashboard > New Item > Freestyle Project(**name : CICD_Pipeline**)
9. Source Code Management > Git > Repo URL >**your github link to be deployed**
   Ex:- https://github.com/subu123321/hello-world.git
10. Branch specifier: ***/master**
11. Build Now
12. Build steps > Build Environment > **Invoke top-level Maven targets**
    Maven version: **maven_home**
    Goals: **clean compile package**
13. Save
14. Terminal Commands If tomcat not installed
    - **sudo su root**
    - **sudo useradd -s /bin/false -g tomcat -d /opt/tomcat tomcat**
    - **cd**
    - **cd /tmp**
    - **curl -O https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.93/bin/apache-tomcat-9.0.93.tar.gz**
    - **mkdir /opt/tomcat**

- **sudo tar xzvf apache-tomcat-9.0.54.tar.gz -C /opt/tomcat --strip-components=1**
- **cd /opt/tomcat**
- **sudo chgrp -R tomcat /opt/tomcat**
- **sudo chmod -R g+r conf**
- **sudo chmod g+x conf**
- **cd ..**
- **sudo chown -R apsit:apsit tomcat/**
- **cd**
- **cd /opt/tomcat/conf**
- **vi server.xml** Change port to 8090
- **vi tomcat-users.xml** Change username/pass to admin / admin

15. If Tomcat is installed
16. Terminal
    - **cd /opttomcat/bin**
    - **./startup.sh**
17. Go to browser and search **localhost://8090**

Exp6 (To implement Jenkins Master-Slave Architecture with Scaling.)

1. Go to jenkins
2. Click Manage Jenkins > Manage Nodes
3. Select New Node and enter the name of the node in the Node Name field.
4. Select Permanent Agent and click the OK button. Initially, you will get only one option,"Permanent Agent." Once we have one or more slaves you will get the "Copy Existing Node"option. Click Create
5. Terminal
    - **pwd**
    - **sudo su root**
    - **find / -type f -name java** copy jdk path
6. Manage Jenkins > Manage Nodes > Node properties > Environment variables > Value > **paste jdk path**
7. Save,will display the below page with error message. Here Jenkins connect with Slave node using Java Web Start and it needs a port to establish the connection.
8. To configure JNLP port in global security. Now goto
   Manage jenkins > Security >Agents >tcp port: **Fixed > 5000**

9. Terminal
   sudo ufw allow 5000/tcp
10. Again coming back to Jenkins and navigate to Nodes -> agent2 which will display two ways to connect with Agent node.
11. Run the provided commands in your terminal
12. This will establish connection with the configured Slave node.
13. Create a new Freestyle project
14. In configuration > click **Restrict where project can be run >** Label Expression : **agent2**
15. Build Steps > Execute shell > **echo "welcome to master-slave architecture"**
16. Build now
17. Check Console Output

Exp7 (To implement selenium automation.)

1. Launch Mozilla Firefox browser.

2. Open URLhttps://addons.mozilla.org/en-us/firefox/addon/selenium-ide/It will redirect you to the official add-on page of Firefox.

3. Click on "Add to Firefox" button.

4. A pop-up dialog box will be appeared asking you to add Selenium IDE as extension to your Firefox browser.

5. Click on "Add" button.

6. Go to the top right corner on your Firefox browser and Click on that icon to launch Selenium IDE.

7. Click on : Create a new project
8. Rename the project as "selenium_demo".
9. Rename the test case as "javaTpoint_test".
10. Click on the "Start Recording" Button present on the top right corner on the IDE to start recording the test case.

11. Go to your Firefox browser and open URL:www.google.com

12. Search any thing , browse images

13. As a result you will see all the records in Selenium about all of your actions

14. Click on the "Run Current Test" button present on the tool bar menu of the IDE. It will execute all of your interactions with the browser and gives you an overall summary of the executed test script.

Exp8 (To demonstrate container lifecycle using various docker commands)

1. **sudo apt-get install docker**
2. **docker - -version**
3. **sudo su root**
4. **docker images**
5. **docker run -it -d ubuntu**
6. **docker ps**
7. **docker ps -a** (copy container id of the first ubuntu image)
8. **docker exec -it** containerID **bash**

   Ex**: docker exec -t 44251452d54d**

9. **ls**
10. **mkdir exp8**
11. **cd exp8**
12. **touch ubuntu_image.txt**
13. To use mysql
    - **docker pull mysql**
    - **docker images**
    - **docker run --name test-mysql -e MYSQL_ROOT_PASSWORD=strong_password -d mysql**
    - In new terminal
      - **docker ps**
      - **docker exec -it** container-Name **bash**

        Ex: **docker exec -it test-mysql bash**

      - **mysql -u root -p**
      - **show databases**
      - **use** database_name
      - **create table sample(id int , name varchar(20));**
      - **show tables**

Exp9 (To build an image for a sample web application from a docker file using . various docker file instructions)

1. Install docker like exp8
2. **mkdir exp9**
3. **cd exp9**
4. **vim index.html**

   echo “this page belongs to Exp9”

5. **vim Dockerfile**
6. **docker info**
7. **sudo service docker start**
8. **sudo docker build -t exp9 .**
9. **sudo docker images**
10. **sudo docker run -p 8083:80 exp9**
11. Open browser and type **localhost://8083**

Exp10 (Installation of Ansible on top of AWS instance. Configure SSH access to Ansible master/slave and setup ansible host and test the connection.)

1. Login to AWS and go to EC2
2. Create 2 instances:
   - ansible_master
   - ansible_slave
3. Create 2 separate terminals for Master and slave each
4. ***Master Terminal:***
   - **ssh -i “ansible_key.pem” ubuntu@**public ipv4 address of ansible_master
   - **sudo su root**
   - **ansible --version**
   - **ssh keygen -t rsa**
   - **cd /root/.ssh**
   - **ls**
   - **cat id_rsa.pub** (copy the entire key)
   - **nano /etc/ansible/hosts**
   - Add the ip address of slave in the hosts as

     slave1 ansible_host=<Slave_IP_address> ansible_user=ubuntu

- **ansible slaves -m ping**
- **ansible slave1 -m apt -a “name=git state=present” --become**
- **ansible slave1 -m apt -a “name=nano state=present” --become**

5. ***Slave Terminal:***
    - **ssh -i “ansible_key.pem” ubuntu@**public ipv4 address of ansible_slave
    - **sudo su root**
    - **ansible --version**
    - **echo "**paste the copied key from master**" >> /root/.ssh/authorized_keys**
    - **git --version**
    - **nano file1.txt**