____

Practical 4

____

Jenkins Practical 4.1
> install
> not reomended option
> port test
> Select JDK fie
> next > install
> yes
> go to chrome search localhost8080
> paste the given path in filemanager
> you will get the password in notepad
> copy that in localhost 8o8o > then choose suggested option
> enter details > username and password as root
> install is done

>new item > freestyle > general > build step > add > apply and save > build > 1 > conole output >

Jenkins Gittest Practical 4.2
new item
> freestyle
> general
> source code management choose "GIT'
> in repository enter the url of your repository
> check the specifer in your github repo and then enter the branch specifier accordingly
> build steps
> add
> enter "node filename"
> apply and save
> click in build now
> click in now
> console output

____

*
*
*
*
*
*
*

____

Practical 5

____

Jenkins Pipeline Practicla 5.1 (Without github - Scripted)
> Dashboard
> Manage Jenkkins
> Plugins
> available plugiins
> search "pipelines"
> select "Pipeline: Deprecated Groovy LibrariesVersion 612.v55f2f80781ef"
> install
> after installed go to dashboard > new item > pipeline > ok > genral
> Find the pipeline section there you will find script
paste the below code

```
pipeline {
agent any
stages {
stage("build") {
steps {
echo 'building the application..'
}
}
stage("test") {
steps {
echo 'testing the application..'
}
}
stage("deploy") {
steps {
echo 'deploying the application..'
}
}
}
}
```

> Apply and Save
> click on buildnow
So far what we have done can be considered as basic pipeline

Jenkins Pipeline Practicla 5.2 (with Git - Descriptive)
-------Note: Before starting this, make a repository in your Github for this particular Practical
Then create a new file and paste the following code:

```
--------------------------------------------------------------------------------
pipeline {
agent any
stages {
stage('Build') {
steps {
echo 'Building the project...'
}
}
stage('Test') {
steps {
echo 'Running tests...'
}
}
stage('Deploy') {
steps {
echo 'Deploying to production...'
}
}
}
}
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
save the file in your repository
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
>Dashboard
> Manage Jenkins
> Pllugins
> available plugins
> search, select and install "Pipeline Maven IntegrationVersion
1396.veb_f07b_2fc1d8", "Pipeline Utility StepsVersion 2.16.2", "Maven
IntegrationVersion 3.23"
> Dashboard and then new item
> in general select Github project and in build triggers select "Build whenever a
SNAPSHOT dependency is built" "GitHub hook trigger for GITScm polling" checkboxes
> in Pipleline section change the defination select the SCM option, below that in
SCM select Git option
> in repository URL copy/paste the URL of you repository and make sure to add
".git" at the end of the URL of your repository
> make sure you have selected the branch which is matching with your selected
repository branch
> in script path enter the name of file which is in your selected repository
> apply, save and build
_________________________________________________________________________________
_________________________________________________________________________________
__
*
```

*
*
*
*
*
____
__
Master/Slave Jenkins Practical 10
____
__
> dashboard
> Manage Jenkins
> Node, then create new node
> name the node and select the permanent option
> in remote root directory enter "Jenkins/Workspac
> then copy the script of windows and paste that in your cmd and then wait for the connecting prompt.
-----note: if there is failure in connection follow the steps: >Dashboard> manage jenkins
> Security
> make sure the TCP port for inbound is set to "Random", apply and save
this should solve the connection error---
> Create a folder named workspace in your Jenkins folder loacted in your system and copy-paste the jar file in Jenkins folder
> Masternode is connected now
> Dashboard
> new item and choose pipeline
> enter the code in script section:

```
-------------------------------------------------------------
node('Name of your node'){
stage('Build') {
 bat '''echo build steps'''
}
stage('Test') {
 bat '''echo test steps'''
}
}
------------------------------------------------------------
```

> apply. save and Build Now
____
_____
*
*
*

*
*
*
____

Ansible Practical 9
____

>install Ubuntu app from Microsoft Store
>after installing the ubunto app, open and it will install
----------Note: if you are getting any error related to unable to install linux or The Windows Subsystem for Linux has not been enable
                  Then goto >Control Panel > Programs > Pragrams and features > Turn windows features on and off > there make sure "Windows Subsystem for Linux" is check, if not then check it and select OK
if will ask to restart the PC, press OK, this should solve the error---

```
> now Open Ubuntu and copy paste the following command
 > 'sudo apt-get update' -> it will ask for password, enter the password to proceed ahead
 > 'sudo apt-get install -y software-properties-common'
 > 'sudo apt-add-repository ppa:ansible/ansible'
 > 'sudo apt-get install -y ansible' -> this will install ansible
 > 'mkdir tuts'> creats the folder named tuts
 > 'cd tuts' > change working directory to tuts
 > 'mkdir -p ansible1'
 > 'cd ansible1'
 > after you are done with the above commands open VScode
 > Open WSL extension or install and open WSL extension
 > connect to WSL by clicking on the blue which as icon >< on the bottom left of VS screen and choose "Connect to WSL"
 > Open a folder then click on tuts
 > Now in ubuntu enter the command 'mkdir inventory'
 > Go to VS code and create file a file inside the inventory named 'Inventory.ini'
 > copy the following code inside the .ini file
```

```
[webservers]
web1.example.com
web2.example.com
[dbservers]
one.example.com
two.example.com
three.example.com
```

 > In ubunto enter command 'cd ..' -> this should take you from 'ansible1' to

'tuts' directory
 > Enter commmand 'ansible-config init > ansible.cfg'
 > again go to ansible1 directory by using command 'cd ansible1'
 > make a directory named playbook by using command 'mkdir playbook'
 > Now go to VS code and in explorer select the playbook and create a file inside playbook folder, file named testplaybook.yml
 > Enter the following code inside the testplaybook.yml

```
---
- name: Update web servers
hosts: webservers
remote_user: root
tasks:
- name: Ensure apache is at the latest version
ansible.builtin.yum:
name: httpd
state: latest
- name: Write the apache config file
ansible.builtin.template:
src: /srv/httpd.j2
dest: /etc/httpd.conf
- name: Update db servers
hosts: databases
remote_user: root
tasks:
- name: Ensure postgresql is at the latest version
ansible.builtin.yum:
name: postgresql
state: latest
- name: Ensure that postgresql is started
ansible.builtin.service:
name: postgresql
state: started
```

And we are done with this practical