1. If you were able to push the code in your branch to the given repo then you answered well.

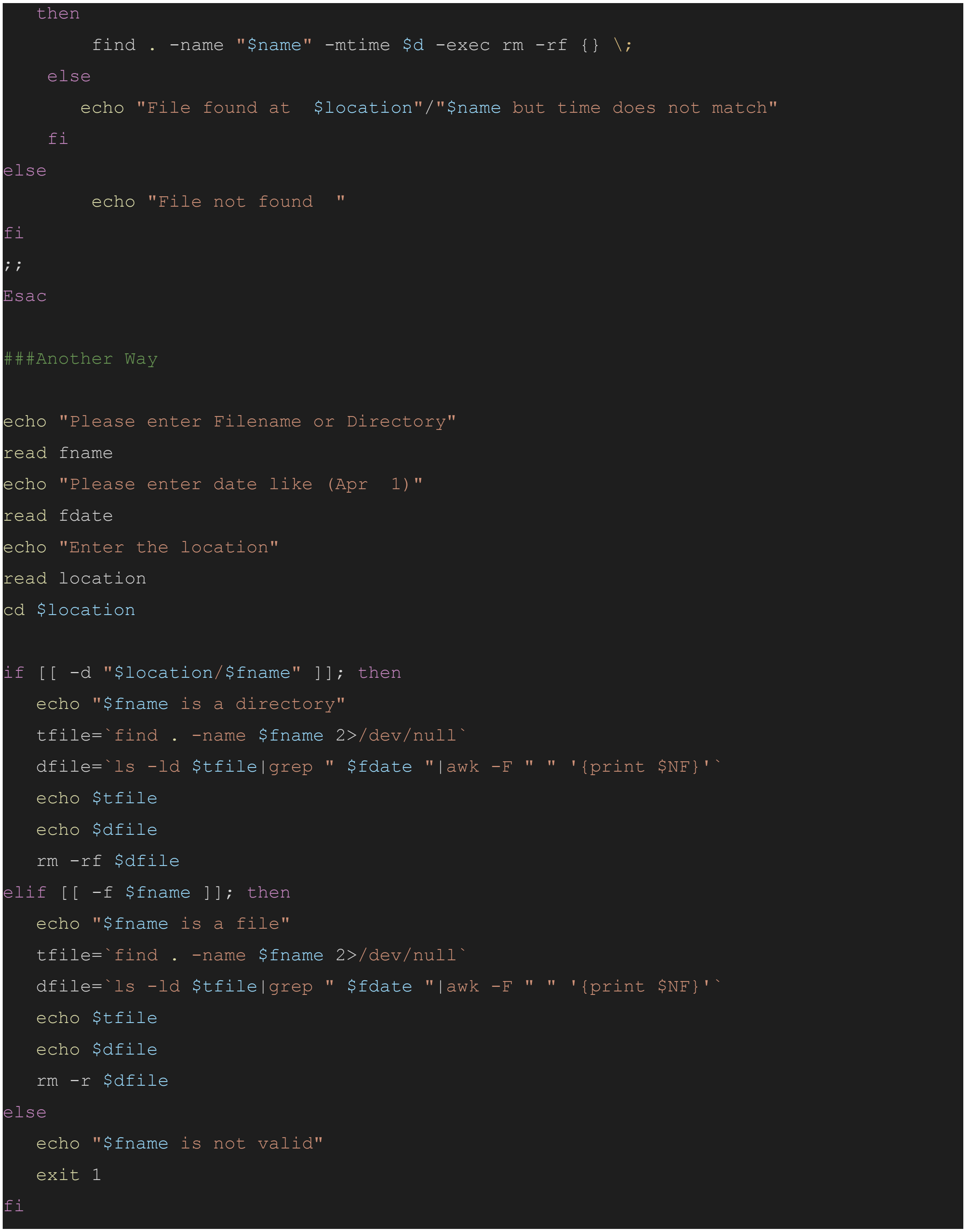
2. Script

```
#!/bin/bash

echo "Ente the Type "Directory" or "File" ,as per your search"
read DirectoryORFile
echo "Type "Directory" or "File" ,as per your search"
read name
echo "Enter the location of File or Directory"
read location
echo "Enter the number days created before  "
read d
case "$DirectoryORFile" in
       "Directory")
if [ -d $location/$name ];
 then
        echo "Directory exists at  $location"/"$name"
        echo "it will be deleted if entered time matches"
        cd $location

   if [[ $(find . -name "$name" -mtime $d) ]]
   then
       sudo find . -name "$name" -mtime $d -exec rm -rf {} \;
    else
       echo "Directory found at  $location"/"$name but time does not match"
    fi
else
       echo "Directory not found"
fi
;;
"File")
if [ -f $location/$name ];
then
        echo "File exists at $location"/"$name"
        echo "it will be deleted if entered time matches"
        cd $location

   if [[ $(find . -name "$name" -mtime $d) ]]
```



```
   then
        find . -name "$name" -mtime $d -exec rm -rf {} \;
    else
       echo "File found at  $location"/"$name but time does not match"
    fi
else
        echo "File not found  "
fi
;;
Esac

###Another Way

echo "Please enter Filename or Directory"
read fname
echo "Please enter date like (Apr  1)"
read fdate
echo "Enter the location"
read location
cd $location

if [[ -d "$location/$fname" ]]; then
   echo "$fname is a directory"
   tfile=`find . -name $fname 2>/dev/null`
   dfile=`ls -ld $tfile|grep " $fdate "|awk -F " " '{print $NF}'`
   echo $tfile
   echo $dfile
   rm -rf $dfile
elif [[ -f $fname ]]; then
   echo "$fname is a file"
   tfile=`find . -name $fname 2>/dev/null`
   dfile=`ls -ld $tfile|grep " $fdate "|awk -F " " '{print $NF}'`
   echo $tfile
   echo $dfile
   rm -r $dfile
else
   echo "$fname is not valid"
   exit 1
fi
```

3. Script

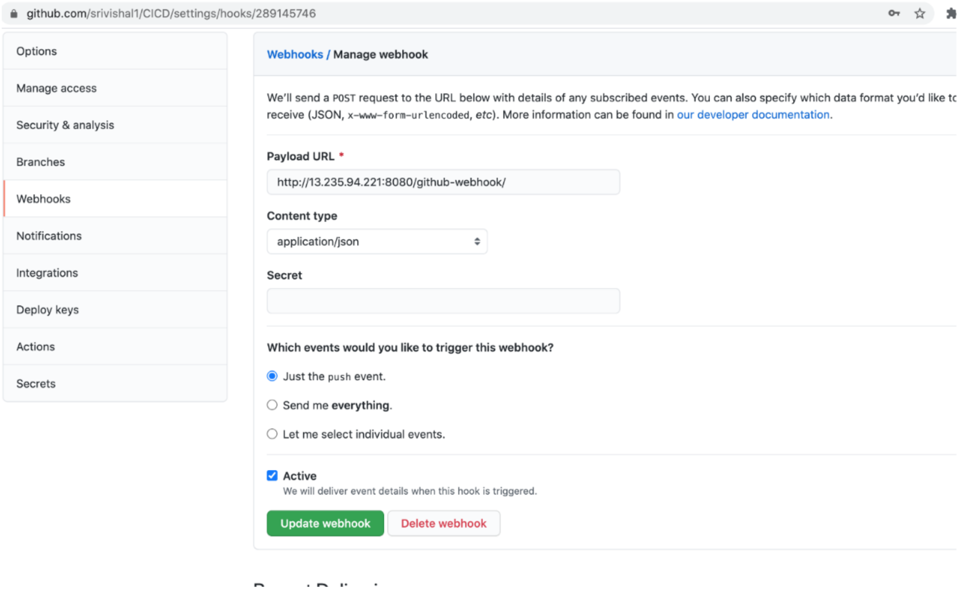
```
#!/bin/bash
FILE=$(grep -hr "/" /etc/passwd  | head -4 > /tmp/file) # This will first 4 users and
redirect to the /tmp/file
FILENAME="/tmp/file"
while IFS=: read -r username password userid groupid home shell
do
echo "username = " "$username"
echo "password = " "$password"
echo "userid = " "$userid"
echo "groupid = " "$groupid"
echo "home = " "$home"
echo "shell = " "$shell"
done < $FILENAME

## OR Another Method
FILE=$(grep -hr "/" /etc/passwd  | head -4 > /tmp/file) # This will first 4 users and
redirect to the /tmp/file
FILENAME="/tmp/file"
for i in `head -n 1 /tmp/file`
do
username=`echo $i|awk -F":" '{print $1}'`
Password=`echo $i|awk -F":" '{print $2}'`
Userid=`echo $i|awk -F":" '{print $3}'`
Groupid=`echo $i|awk -F":" '{print $4}'`
UserInfo=`echo $i|awk -F":" '{print $5}'`
Home=`echo $i|awk -F":" '{print $6}'`
Shell=`echo $i|awk -F":" '{print $7}'`
echo -e "\nusername=$username"
echo "Password=$Password"
echo "Userid=$Userid"
echo "Groupid=$Groupid"
echo "UserInfo=$UserInfo"
echo "Home=$Home"
echo "Shell=$Shell"
```

```
done
```

4. **Steps for Jenkins Job.**

   a) Configure github webhook with Jenkins



We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like tc receive (JSON, x-www-form-urlencoded, *etc*). More information can be found in our developer documentation.

Payload URL *
http://13.235.94.221:8080/github-webhook/

Content type
application/json

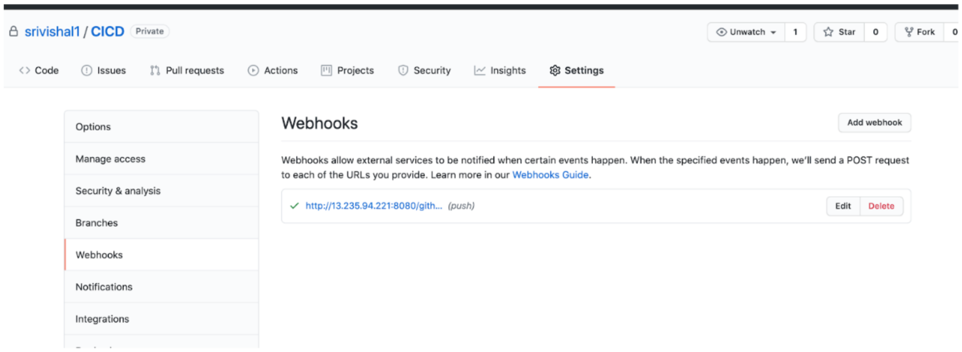
Secret

Which events would you like to trigger this webhook?

● Just the push event.
○ Send me **everything**.
○ Let me select individual events.

☑ Active
We will deliver event details when this hook is triggered.

Validating whether webhook configured properly -



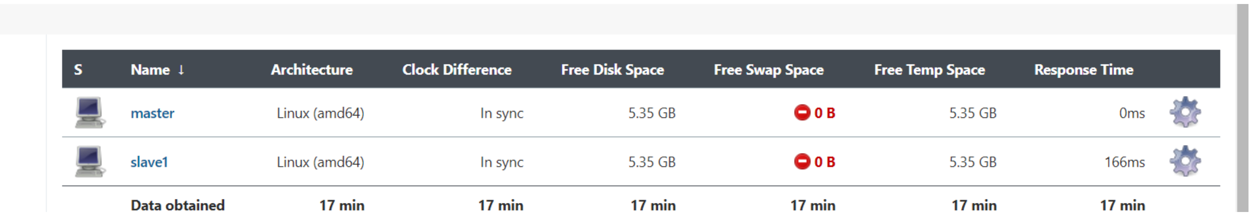
Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

✓ http://13.235.94.221:8080/gith... *(push)*

Setup a New Jenkins Job named as "AssignmentJob"

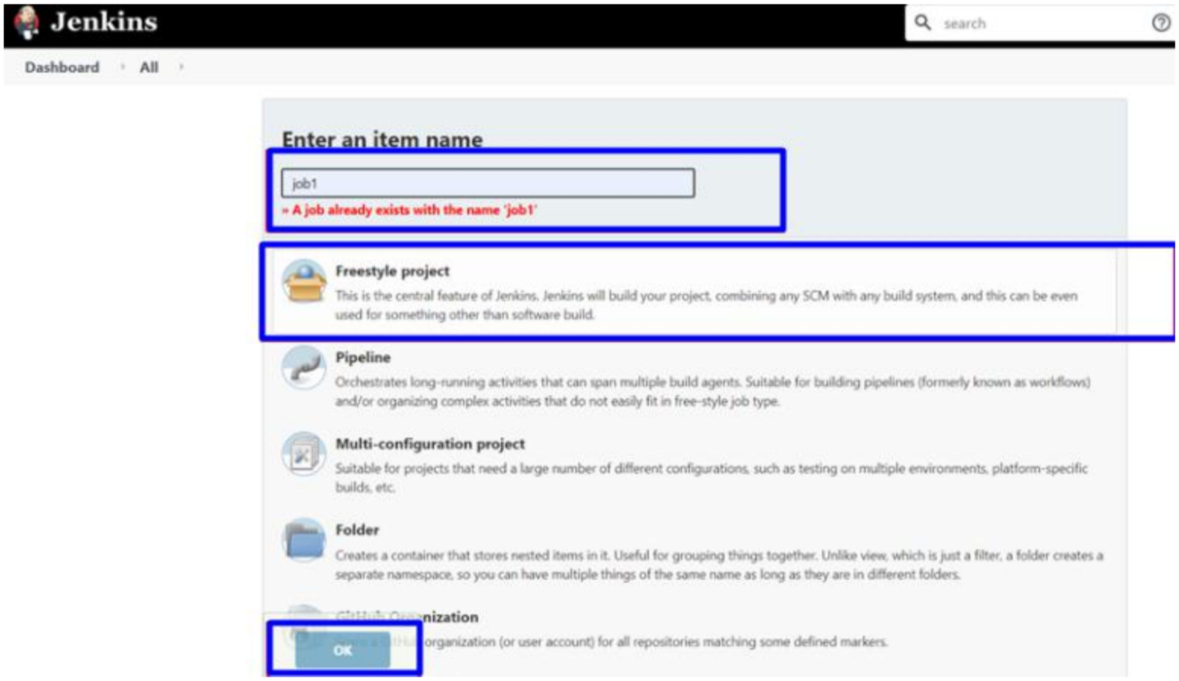
1. Configure Github repository path
2. Select "Restrict where this project can be run" and Select the created slave

b) Create the slave configuration



S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time	
	master	Linux (amd64)	In sync	5.35 GB	⛔ 0 B	5.35 GB	0ms	⚙
	slave1	Linux (amd64)	In sync	5.35 GB	⛔ 0 B	5.35 GB	166ms	⚙
	Data obtained	17 min	17 min	17 min	17 min	17 min	17 min	

c) Create a free style Job with any name.



Enter an item name

job1

» A job already exists with the name 'job1'

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

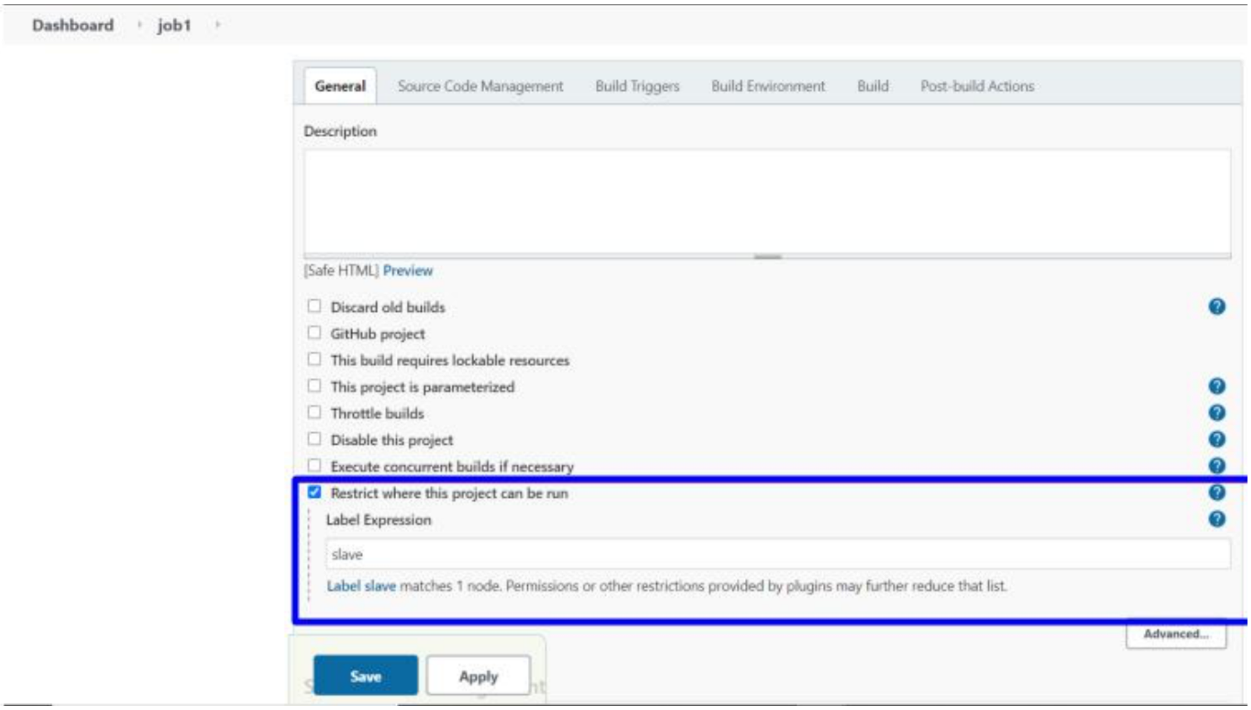
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

d) Configure the job to be bound with a managed selected slave.



Dashboard › job1 ›

Description

[Safe HTML] Preview

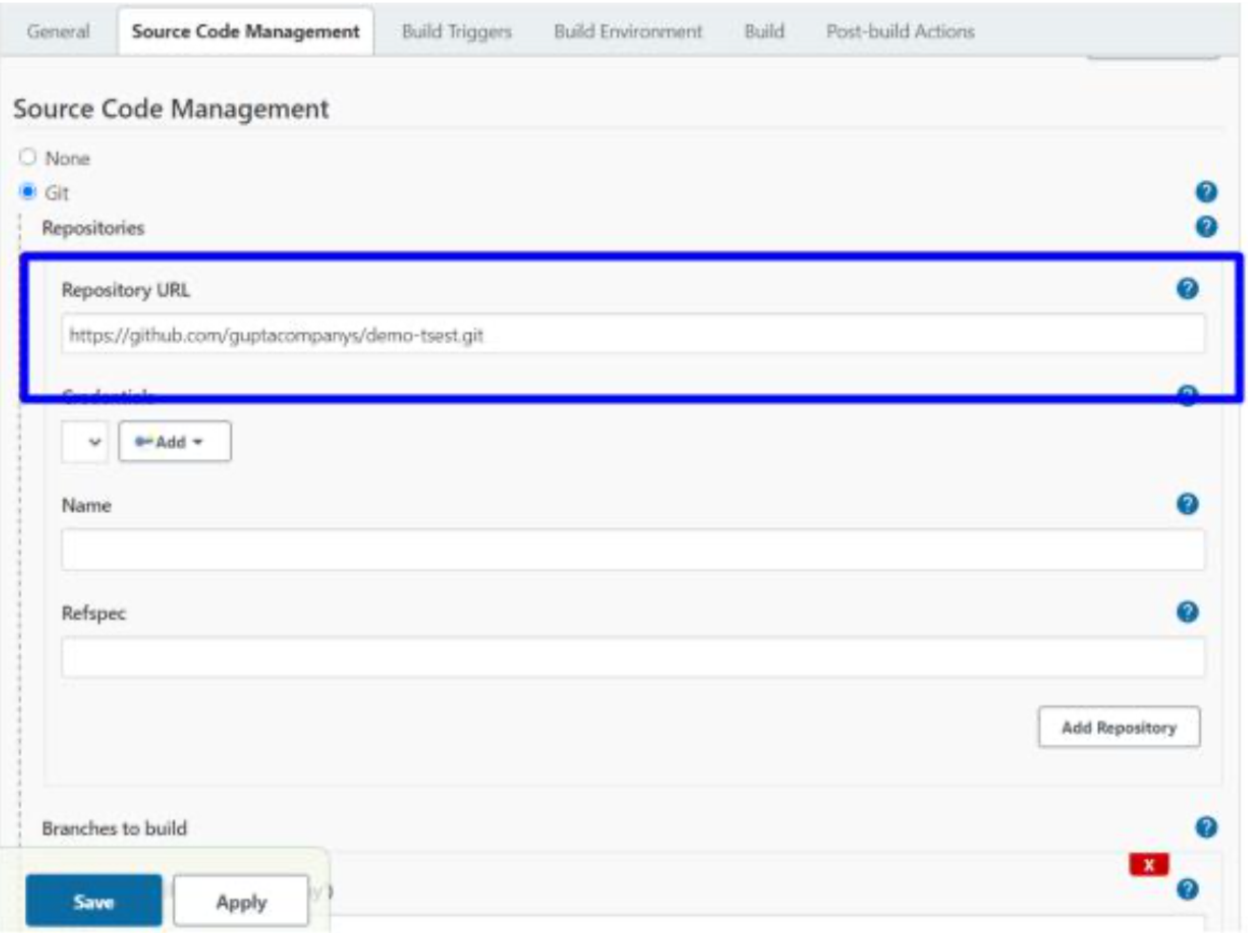
☐ Discard old builds
☐ GitHub project
☐ This build requires lockable resources
☐ This project is parameterized
☐ Throttle builds
☐ Disable this project
☐ Execute concurrent builds if necessary
☑ Restrict where this project can be run

Label Expression

slave

Label slave matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.

e) Configure git repo.



Source Code Management

○ None

● Git

Repositories

Repository URL

https://github.com/guptacompanys/demo-tsest.git

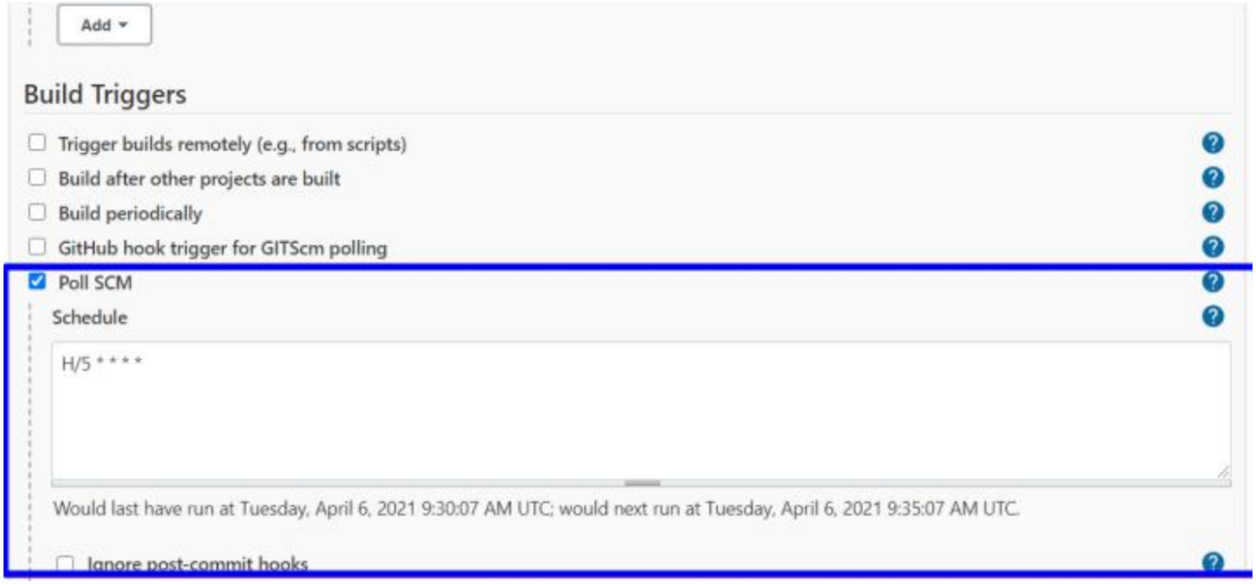
Credentials

Name

Refspec

Branches to build

f) Configure the Poll SCM to check every 5 min job.



Build Triggers

□ Trigger builds remotely (e.g., from scripts)

□ Build after other projects are built

□ Build periodically

□ GitHub hook trigger for GITScm polling

☑ Poll SCM

Schedule

```
H/5 * * * *
```

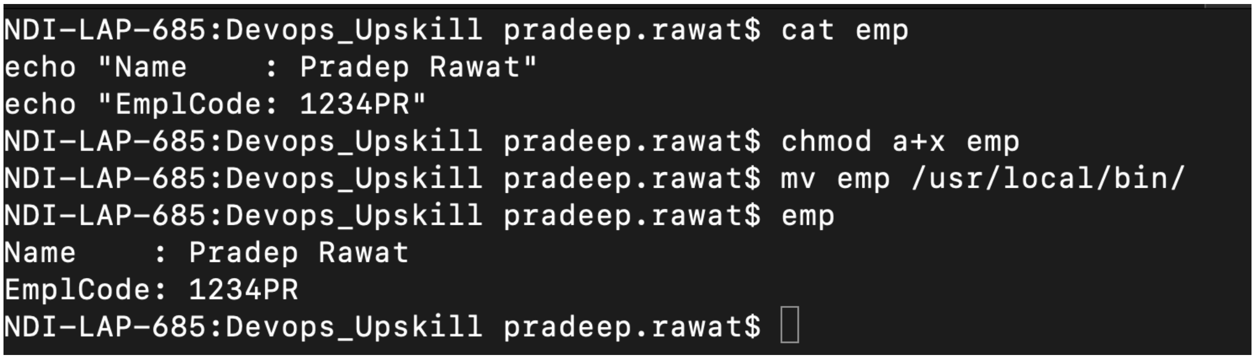
Would last have run at Tuesday, April 6, 2021 9:30:07 AM UTC; would next run at Tuesday, April 6, 2021 9:35:07 AM UTC.

□ Ignore post-commit hooks

5. commands

```
# touch credentials    # create the empty file
# ls -lhrta credentials # check the permissions
# chmod o-r credentials # This removes read permission from others.
# setfacl -m u:<username>:r credentials
# getfacl credentials # To check the new permissions.
```

6.



```
NDI-LAP-685:Devops_Upskill pradeep.rawat$ cat emp
echo "Name    : Pradep Rawat"
echo "EmplCode: 1234PR"
NDI-LAP-685:Devops_Upskill pradeep.rawat$ chmod a+x emp
NDI-LAP-685:Devops_Upskill pradeep.rawat$ mv emp /usr/local/bin/
NDI-LAP-685:Devops_Upskill pradeep.rawat$ emp
Name    : Pradep Rawat
EmplCode: 1234PR
NDI-LAP-685:Devops_Upskill pradeep.rawat$ 
```

Note: There are multiple ways to get these tasks done and few of them I have mentioned. Maybe you have a different approach to achieve it but before choosing any approach you should see which is appropriate for the current scenarios.