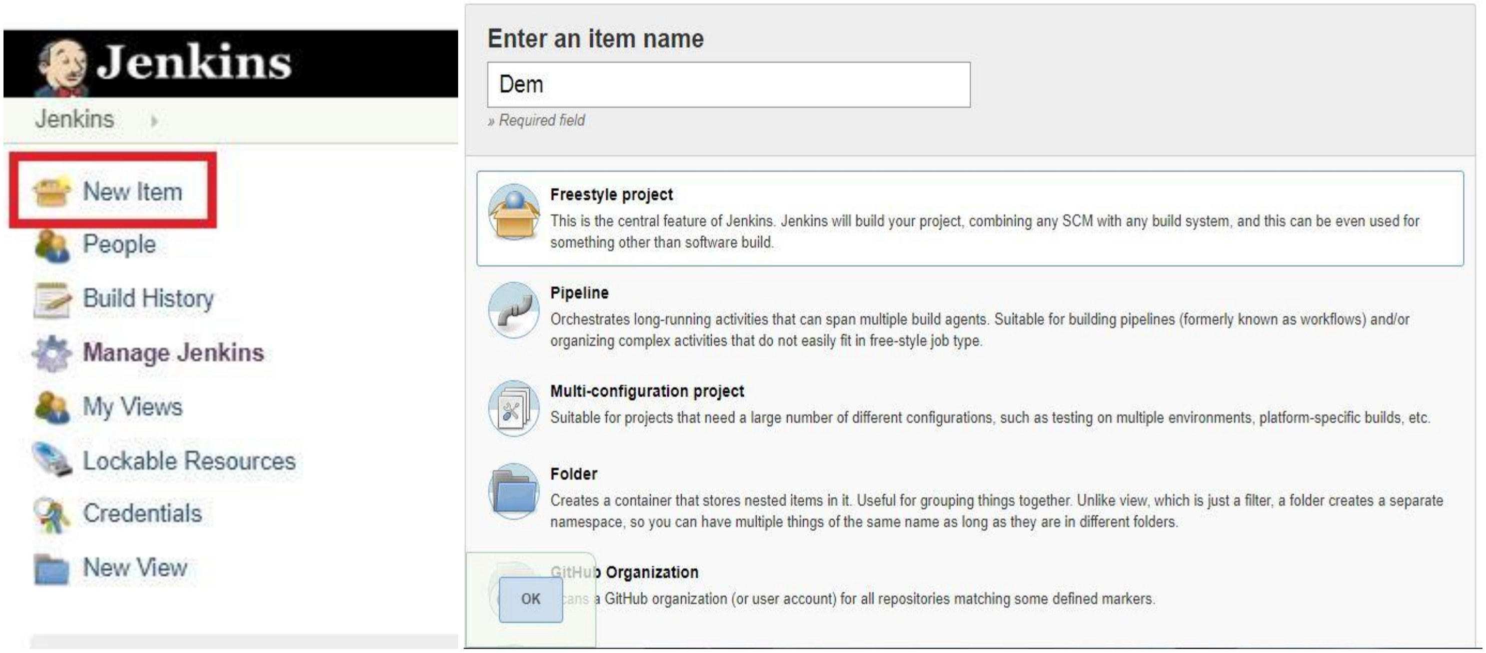
Jenkins LAB	**PARAMETER BUILD**

We can get input from deployer and use the value as variable in project

Step1: Click ‘new item’ in the dashboard. It will redirect to wizard provide ‘item name’ and type of project select ‘Freestyle Project’.



Enter an item name

Dem

» Required field

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.

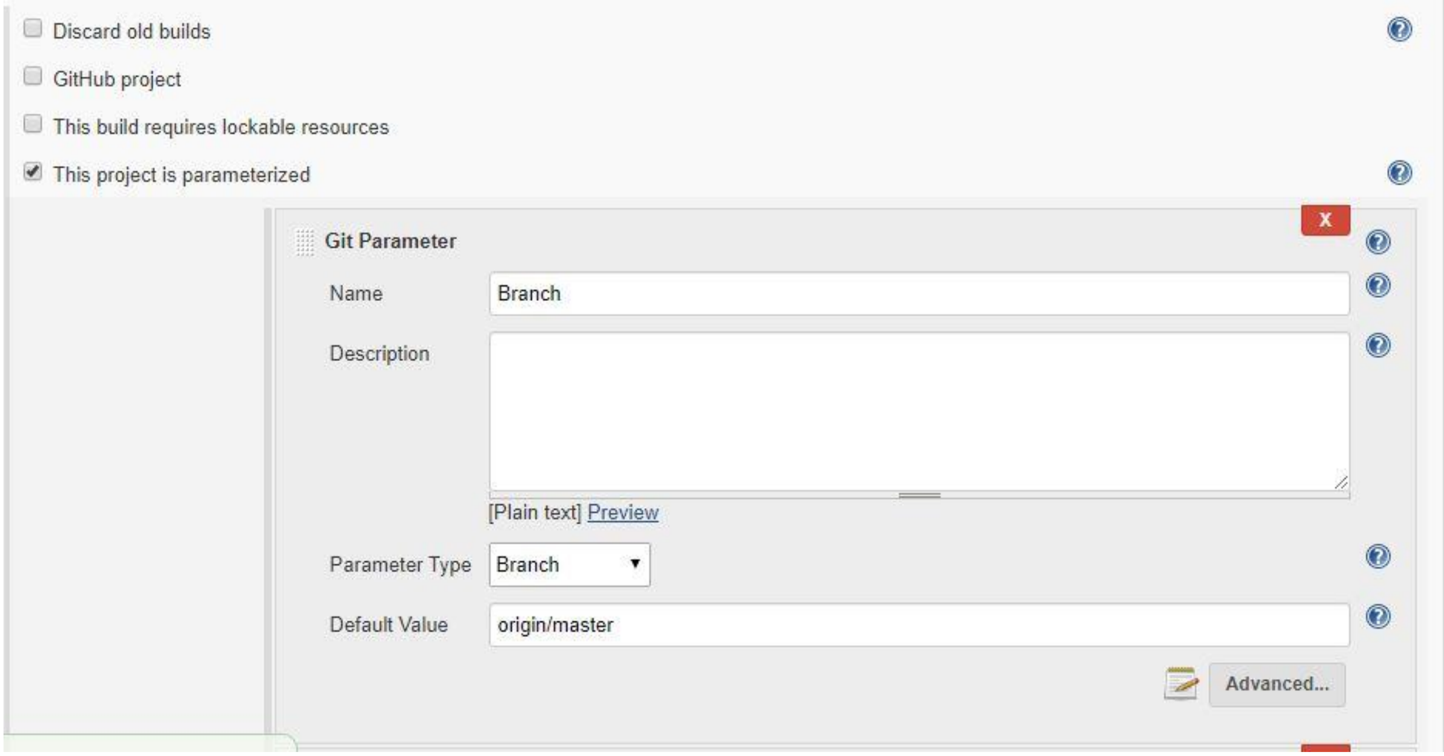
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

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.

Step2: Once the freestyle project wizard open provide the description and add the parameter required here we used Git parameter and Environment.



□ Discard old builds

□ GitHub project

□ This build requires lockable resources

☑ This project is parameterized

Git Parameter

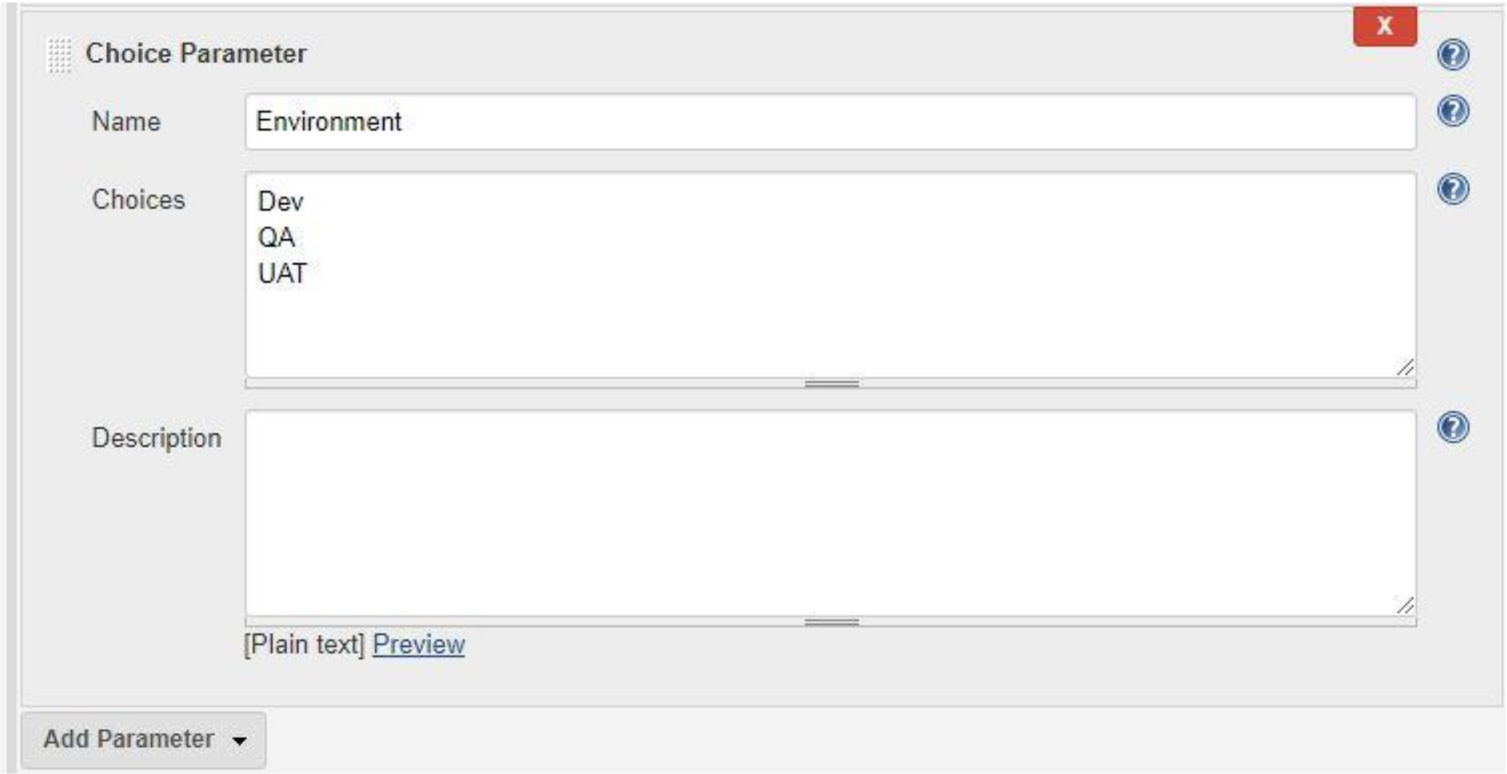
Name: Branch

Description:

[Plain text] Preview

Parameter Type: Branch

Default Value: origin/master



Choice Parameter

Name: Environment

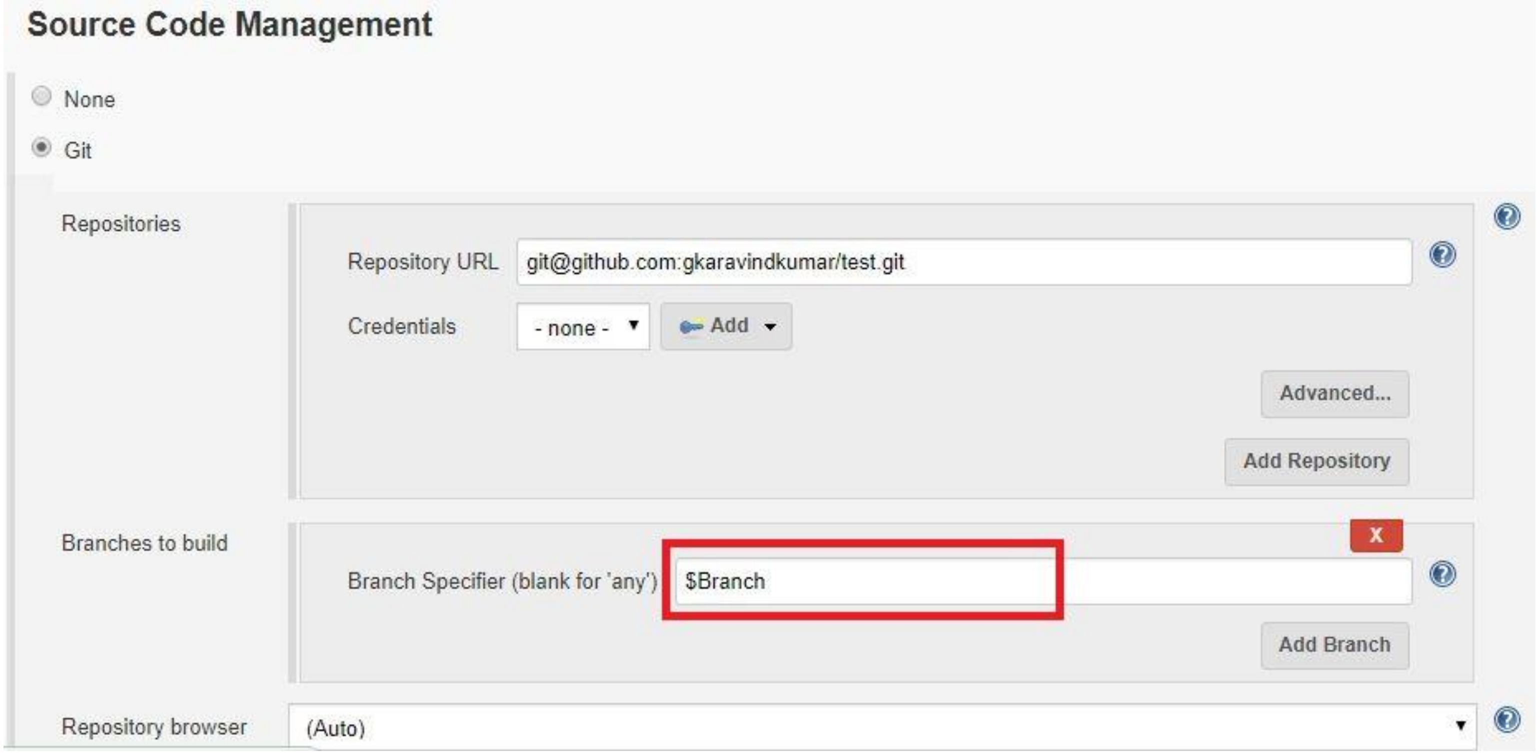
Choices:

```
Dev
QA
UAT
```

Description:

[Plain text] Preview

Step-3: Enter the source code details. (Refer the old freestyle project document for Git integration). Here we need to get the input from git parameter.



Source Code Management

○ None

● Git

Repositories

Repository URL: git@github.com:gkaravindkumar/test.git

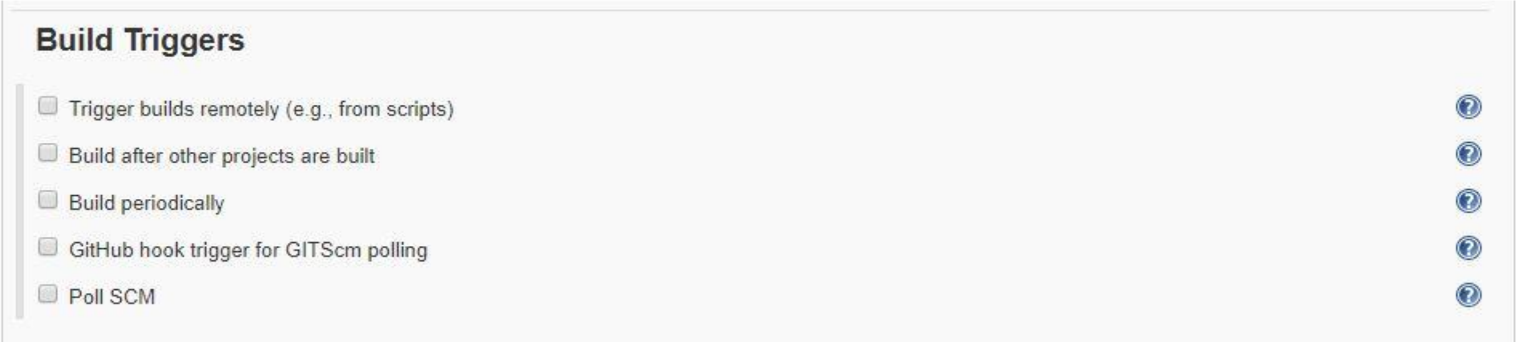
Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): $Branch

Repository browser: (Auto)

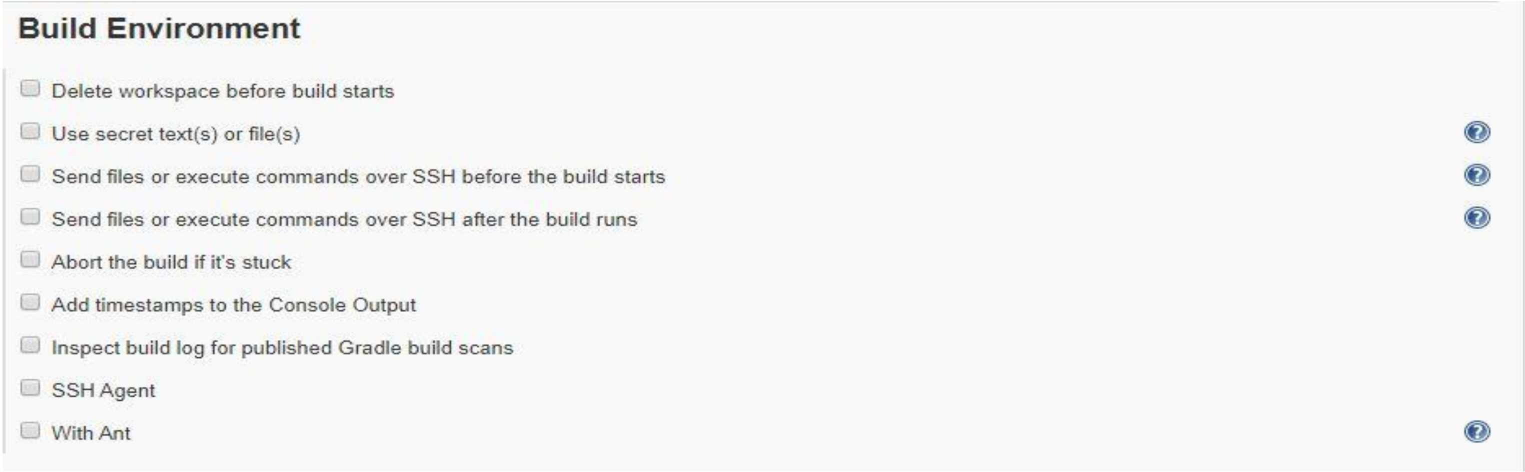
Step-4: Use **Build Triggers** this if needed (we have separate Lab document on Build triggers)



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

Step-5: We can use the **Build Environment** option if needed

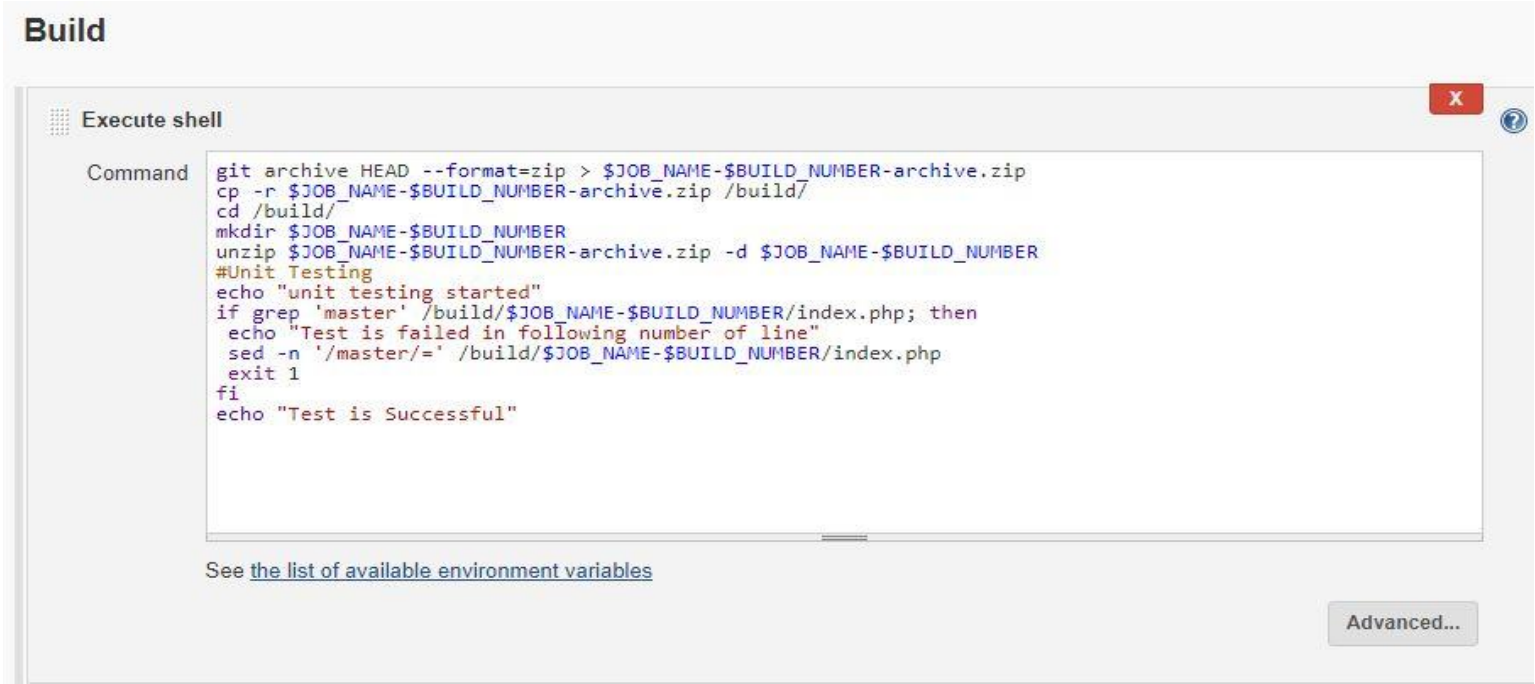


Build Environment

- □ Delete workspace before build starts
- □ Use secret text(s) or file(s)
- □ Send files or execute commands over SSH before the build starts
- □ Send files or execute commands over SSH after the build runs
- □ Abort the build if it's stuck
- □ Add timestamps to the Console Output
- □ Inspect build log for published Gradle build scans
- □ SSH Agent
- □ With Ant

Step-6: In Build option we have multiple options, I select "**Execute shell**" to execute a shell script. This script will bring the build artifacts from GIT to our Jenkins server and place those artifacts in folder **/build** and extract the artifacts then run **Unit Testing**.

Unit testing will fail if index.php have any word as master (demo testing you can add your own script).If **unit testing fails the build will fail**.



Build

Execute shell

Command

```
git archive HEAD --format=zip > $JOB_NAME-$BUILD_NUMBER-archive.zip
cp -r $JOB_NAME-$BUILD_NUMBER-archive.zip /build/
cd /build/
mkdir $JOB_NAME-$BUILD_NUMBER
unzip $JOB_NAME-$BUILD_NUMBER-archive.zip -d $JOB_NAME-$BUILD_NUMBER
#Unit Testing
echo "unit testing started"
if grep 'master' /build/$JOB_NAME-$BUILD_NUMBER/index.php; then
 echo "Test is failed in following number of line"
 sed -n '/master/=' /build/$JOB_NAME-$BUILD_NUMBER/index.php
 exit 1
fi
echo "Test is Successful"
```

See the list of available environment variables

Demo Script here(Make our own script based on your environment)

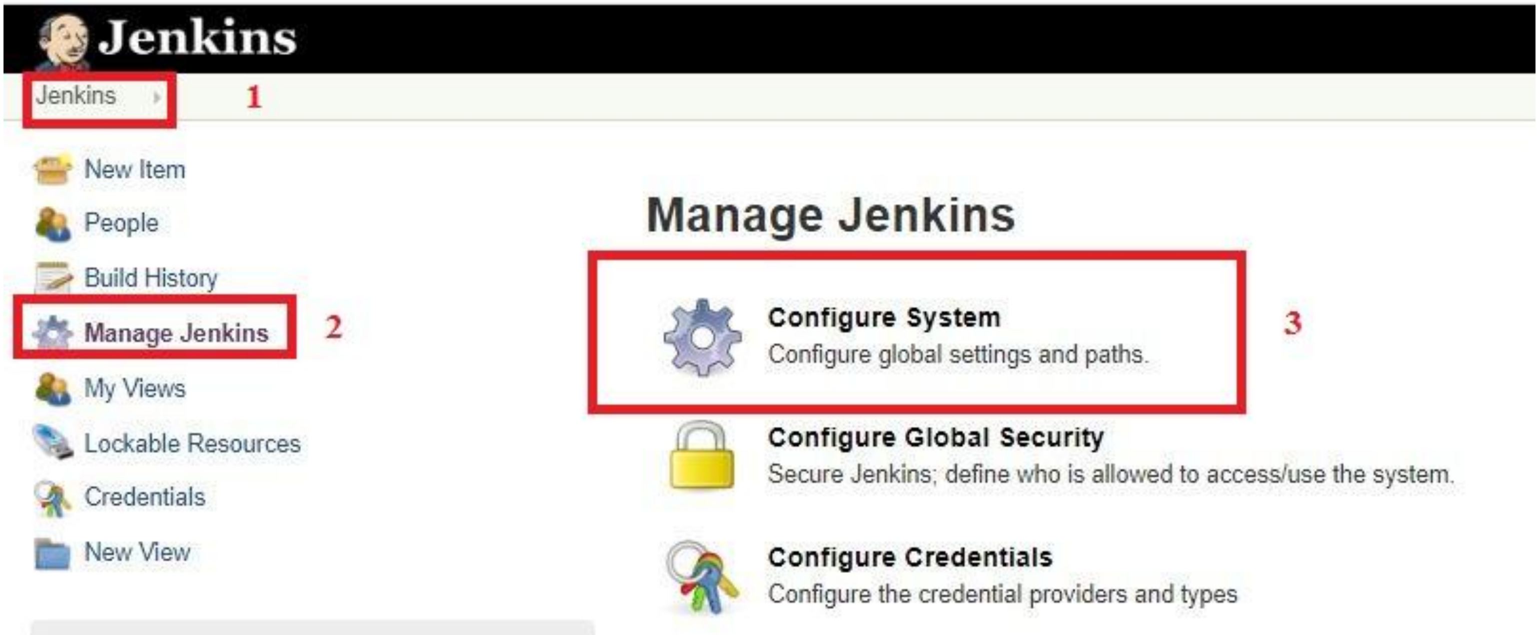
```
git archive HEAD --format=zip > $JOB_NAME-$BUILD_NUMBER-archive.zip
cp –r $JOB_NAME-$BUILD_NUMBER-archive.zip /build
cd /build/
mkdir $JOB_NAME-$BUILD_NUMBER
unzip $JOB_NAME-$BUILD_NUMBER-archive.zip -d $JOB_NAME-$BUILD_NUMBER
#Unit Testing
echo "unit testing started"
if grep 'master' /build/$JOB_NAME-$BUILD_NUMBER/index.php; then
 echo "Test is failed in following number of line"
 sed -n '/master/=' /build/$JOB_NAME-$BUILD_NUMBER/index.php
 exit 1
fi
echo "Test is Successful"
```

Step-7: Moving forward to deployment we need to add hosts/server in Jenkins (web servers where we are going to do the deployments).

To Add ssh host/ Server we need to add plugins (Publish over ssh).Take a look at Jenkins environment setup document for adding plugins.

To add ssh host install Publish over ssh plugin.

Go to Jenkins→ Manage Jenkins→ Configure systems



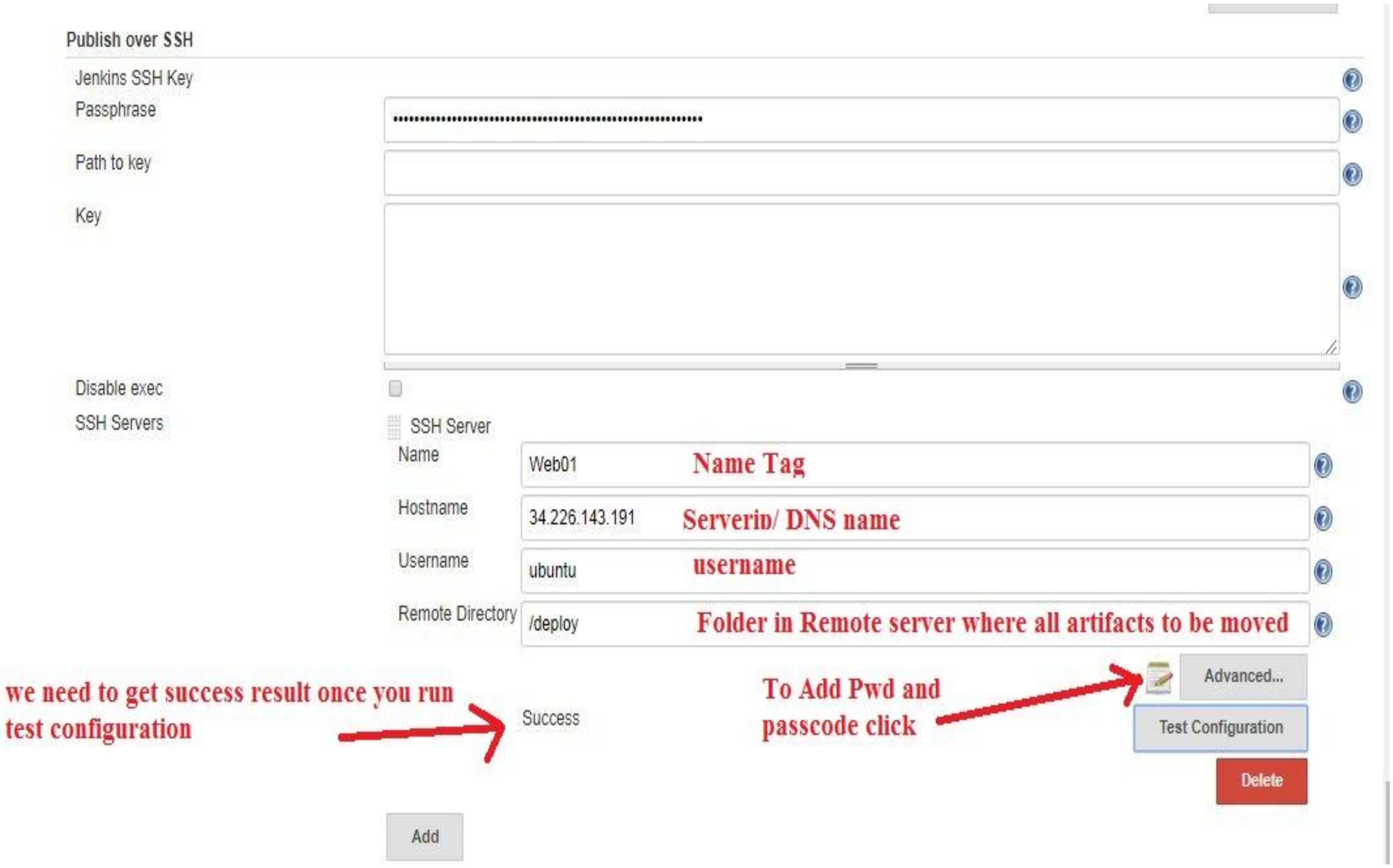
Manage Jenkins

Configure System
Configure global settings and paths.

Configure Global Security
Secure Jenkins; define who is allowed to access/use the system.

Configure Credentials
Configure the credential providers and types

Go to the bottom of Jenkins page you will see **Publish over SSH**



Publish over SSH

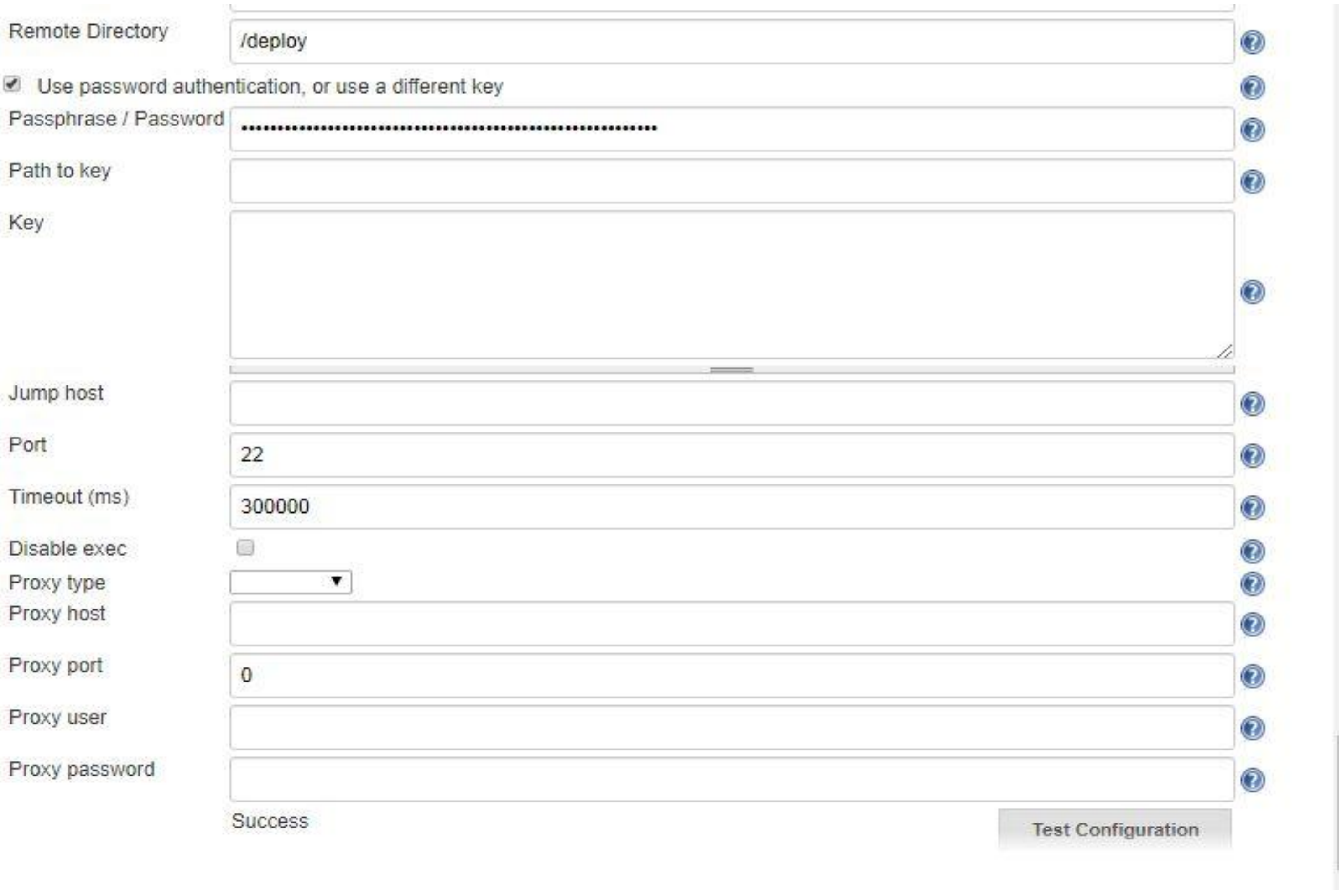
Jenkins SSH Key	
Passphrase	••
Path to key	
Key	
Disable exec	□
SSH Servers	SSH Server

Name	Web01	Name Tag
Hostname	34.226.143.191	Serverip/ DNS name
Username	ubuntu	username
Remote Directory	/deploy	Folder in Remote server where all artifacts to be moved

we need to get success result once you run test configuration

Success

To Add Pwd and passcode click

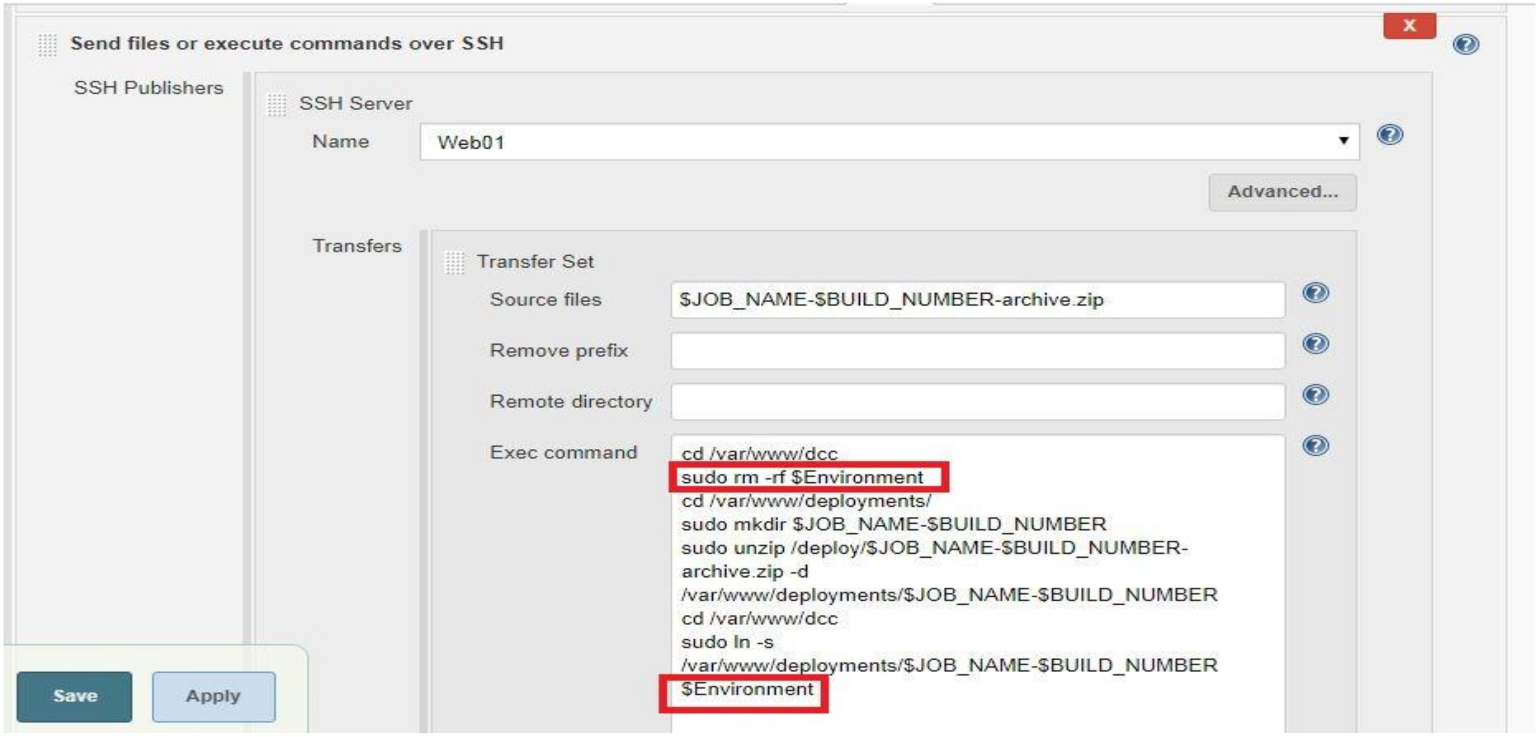


Remote Directory	/deploy
☑ Use password authentication, or use a different key	
Passphrase / Password	••
Path to key	
Key	
Jump host	
Port	22
Timeout (ms)	300000
Disable exec	□
Proxy type	
Proxy host	
Proxy port	0
Proxy user	
Proxy password	

Success

Step-8: Once you added the server in configure systems then return to your Build plan and now in Build options select "**Send files or execute commands over ssh**". You will see the host added there in scroll down list, select your server and other options and "**Execute shell**" which should be executed in remote server.

In the script we need to get the value from Environment parameter so that the script will create/deploy the environment.



Send files or execute commands over SSH

SSH Publishers

SSH Server

Name: Web01

Transfers

Transfer Set

Source files: JOB_NAME-BUILD_NUMBER-archive.zip

Remove prefix:

Remote directory:

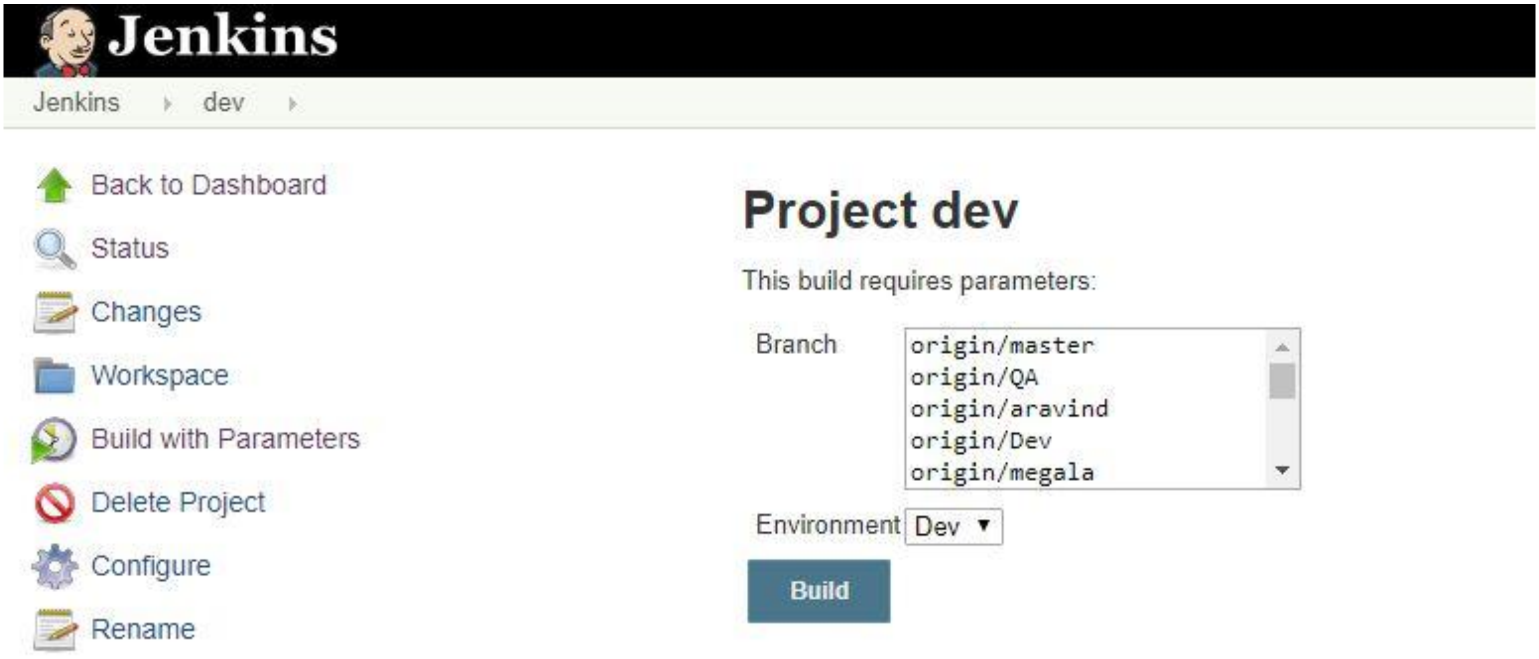
Exec command:

```
cd /var/www/dcc
sudo rm -rf $Environment
cd /var/www/deployments/
sudo mkdir $JOB_NAME-$BUILD_NUMBER
sudo unzip /deploy/$JOB_NAME-$BUILD_NUMBER-
archive.zip -d
/var/www/deployments/$JOB_NAME-$BUILD_NUMBER
cd /var/www/dcc
sudo ln -s
/var/www/deployments/$JOB_NAME-$BUILD_NUMBER
$Environment
```

```
cd /var/www/dcc
sudo rm -rf $Environment
cd /var/www/deployments/
sudo mkdir $JOB_NAME-$BUILD_NUMBER
sudo unzip /deploy/$JOB_NAME-$BUILD_NUMBER-archive.zip -d /var/www/deployments/$JOB_NAME-
$BUILD_NUMBER
cd /var/www/dcc
sudo ln -s deployments/$JOB_NAME-$BUILD_NUMBER $Environment
```

The above script will extract the source artifacts from **/deploy** to **/deployments/JOB_NAME-BUILD_NUMBER** and then create simlink (softlink) **$Environment** inside **dcc** which pointing to **/deployments/JOB_NAME-BUILD_NUMBER** so that current version will be loaded in site.

Now click Build with Parameter option and run the build



Project dev

This build requires parameters:

Branch
origin/master
origin/QA
origin/aravind
origin/Dev
origin/megala

Environment Dev