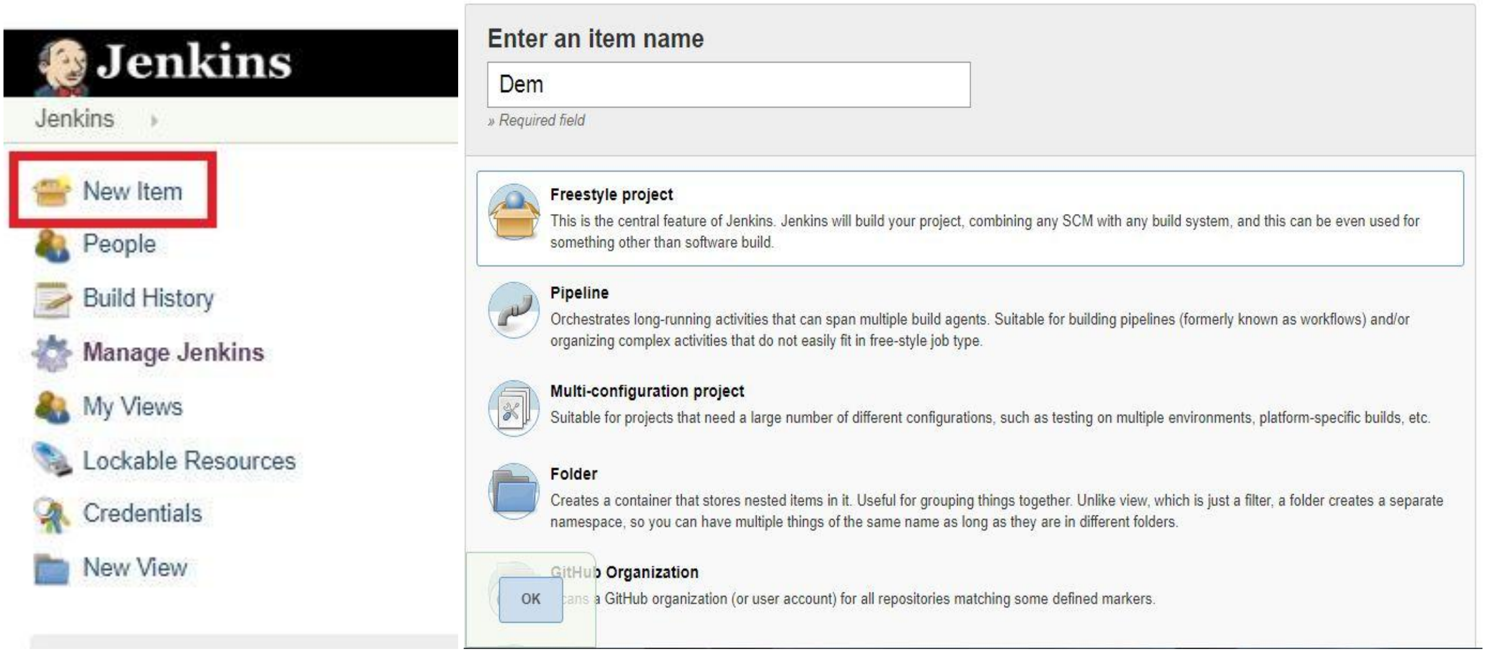
**Step1**: Click ‘new item’ in the dashboard. It will redirect to wizard provide ‘item name’ and type of project select ‘Freestyle Project’.



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.

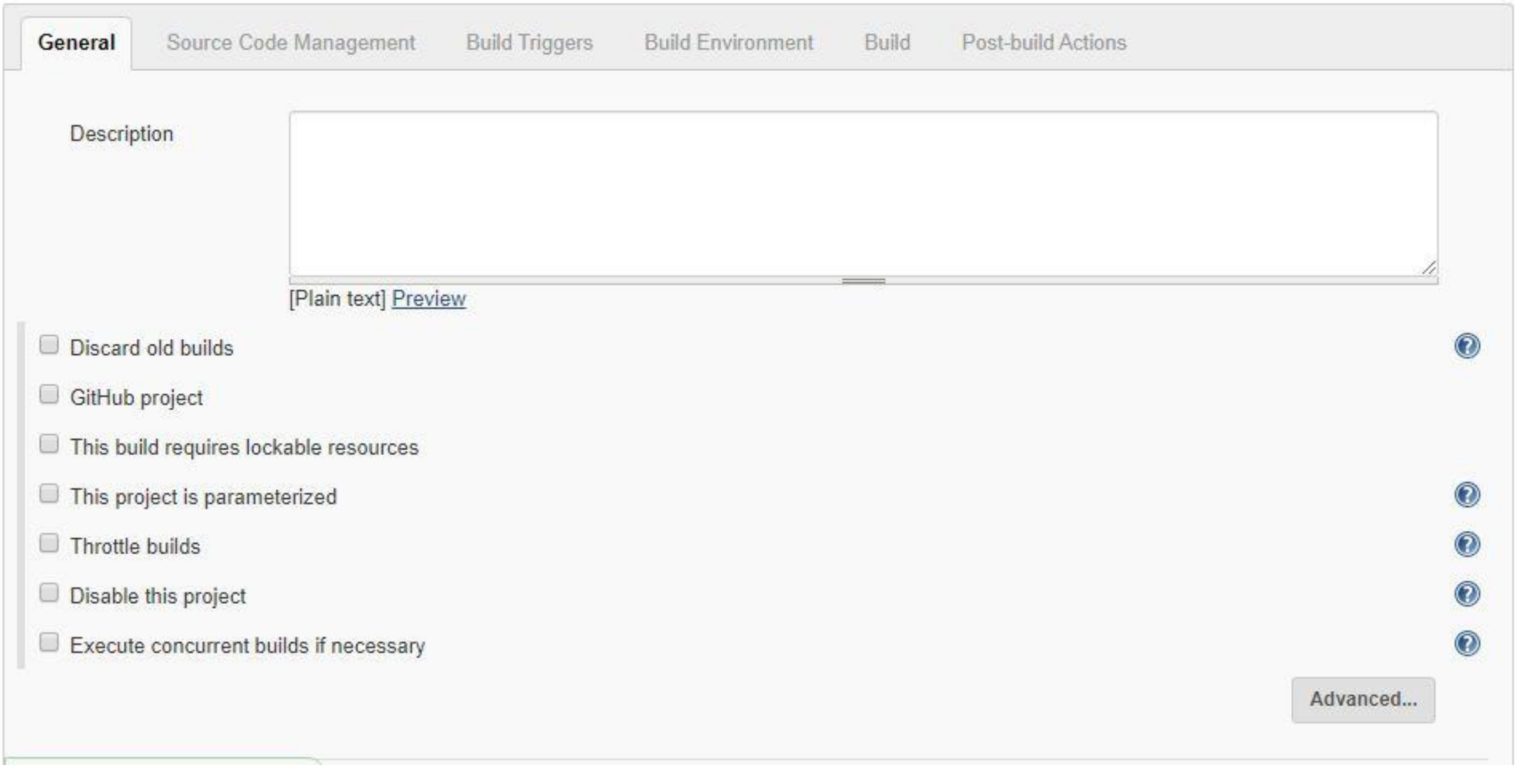
**Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**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.



Description

[Plain text] Preview

☐ Discard old builds

☐ GitHub project

☐ This build requires lockable resources

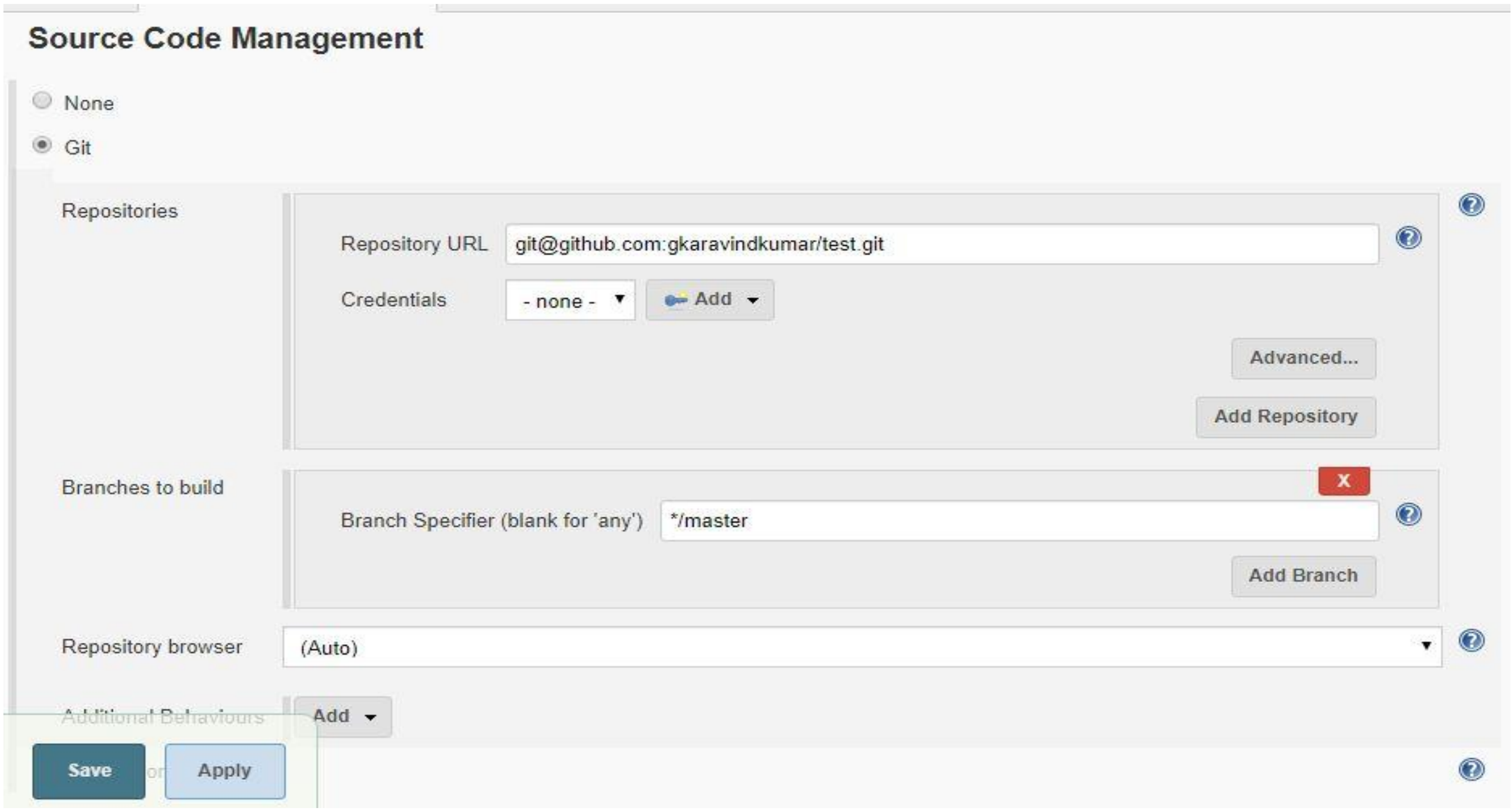
☐ This project is parameterized

☐ Throttle builds

☐ Disable this project

☐ Execute concurrent builds if necessary

**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

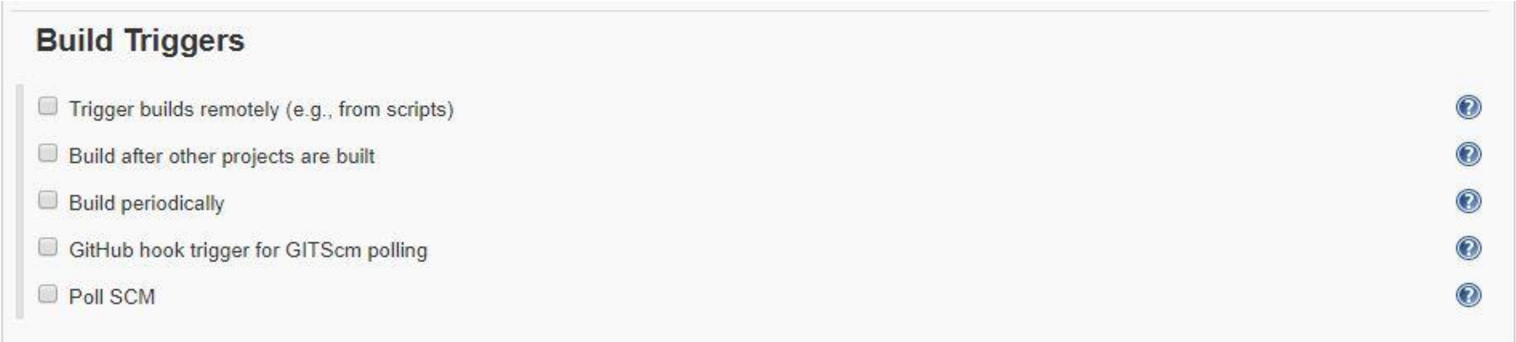
Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */master

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)

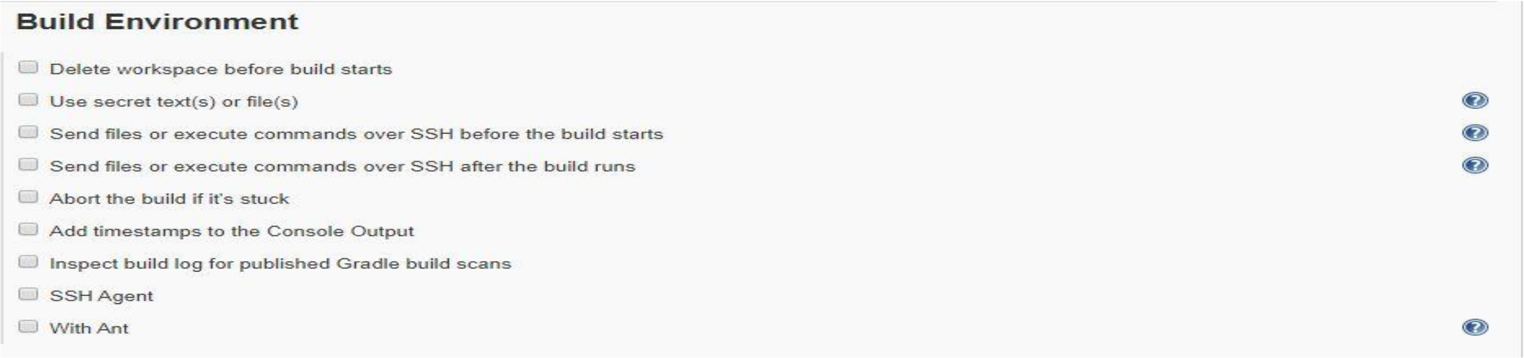
□ Build after other projects are built

□ 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

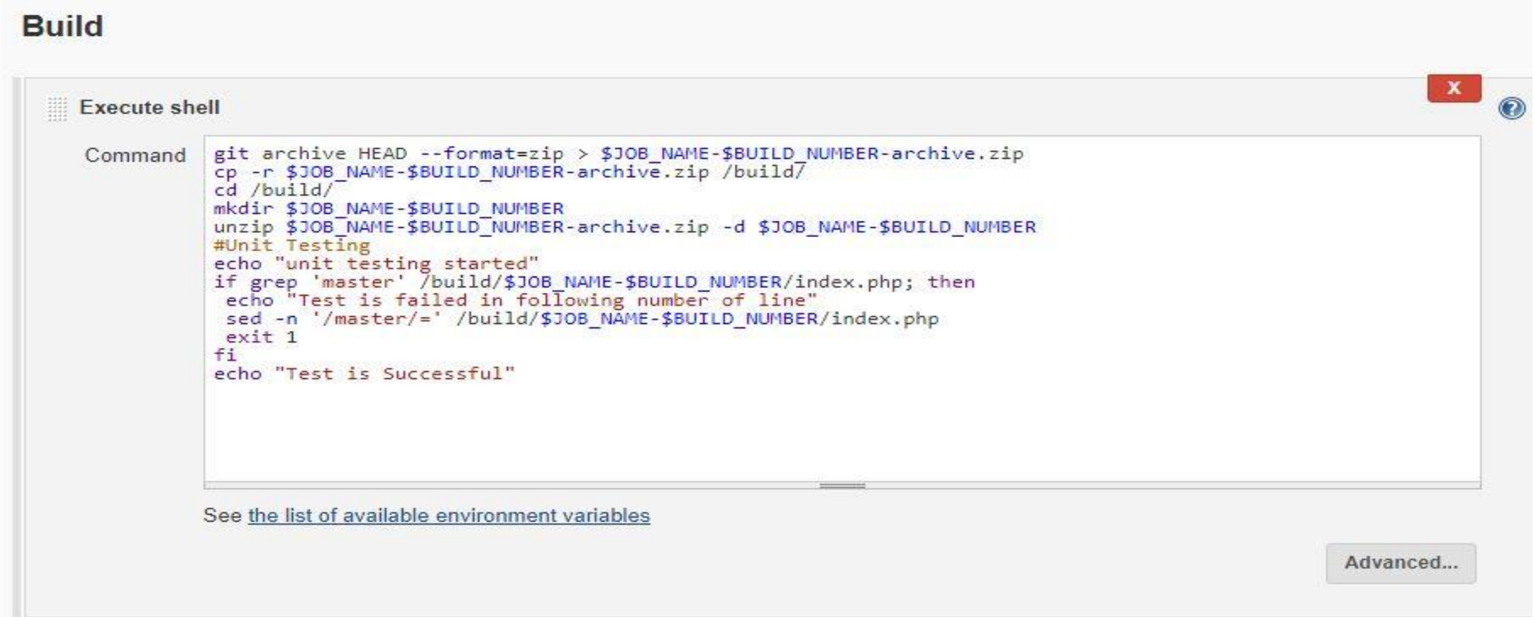
□ Inspect build log for published Gradle build scans

□ 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**.



```
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"
```

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

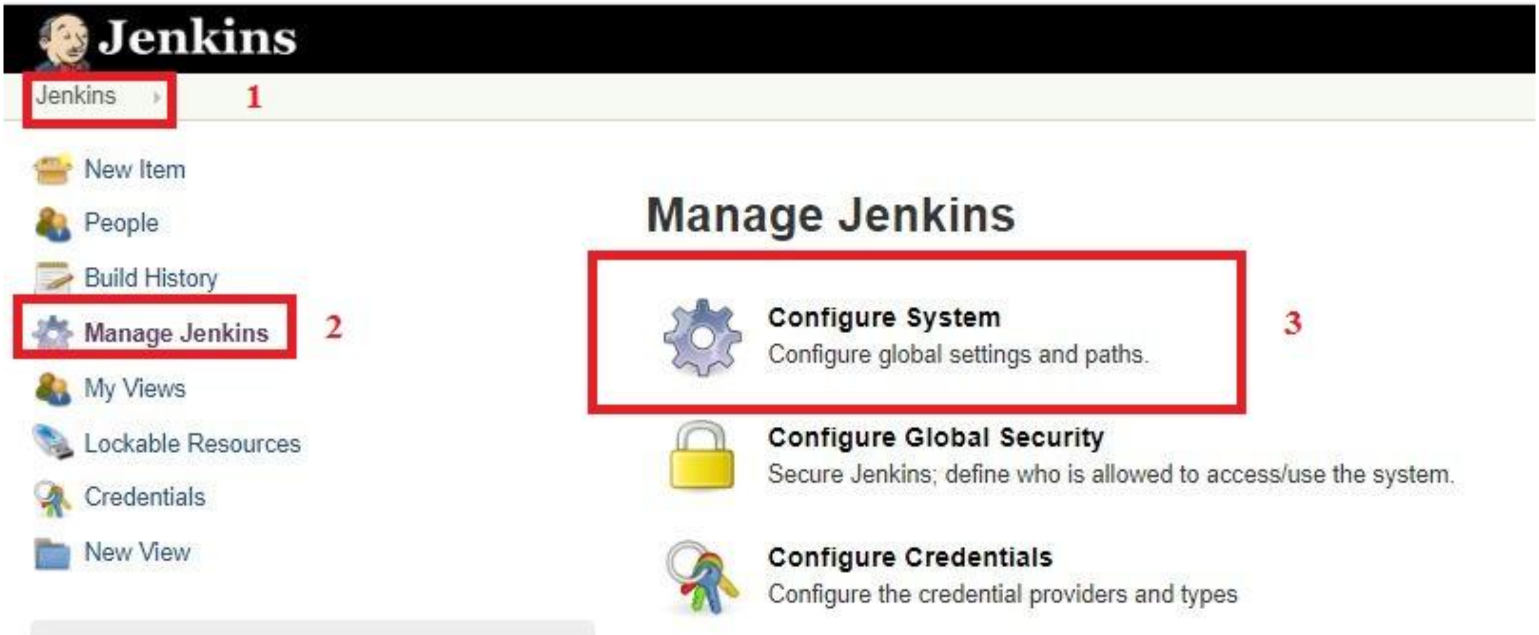
```
git archive HEAD --format=zip > $JOB_NAME-$BUILD_NUMBER-archive.zip
cp –r $JOB_NAME-$BUILD_NUMBER-archive.zip /build
cd /build/
mkdir $JOB_NAME-$BUILD_NUMBER
unzip $JOB_NAME-$BUILD_NUMBER-archive.zip -d $JOB_NAME-$BUILD_NUMBER
#Unit Testing
echo "unit testing started"
if grep 'master' /build/$JOB_NAME-$BUILD_NUMBER/index.php; 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



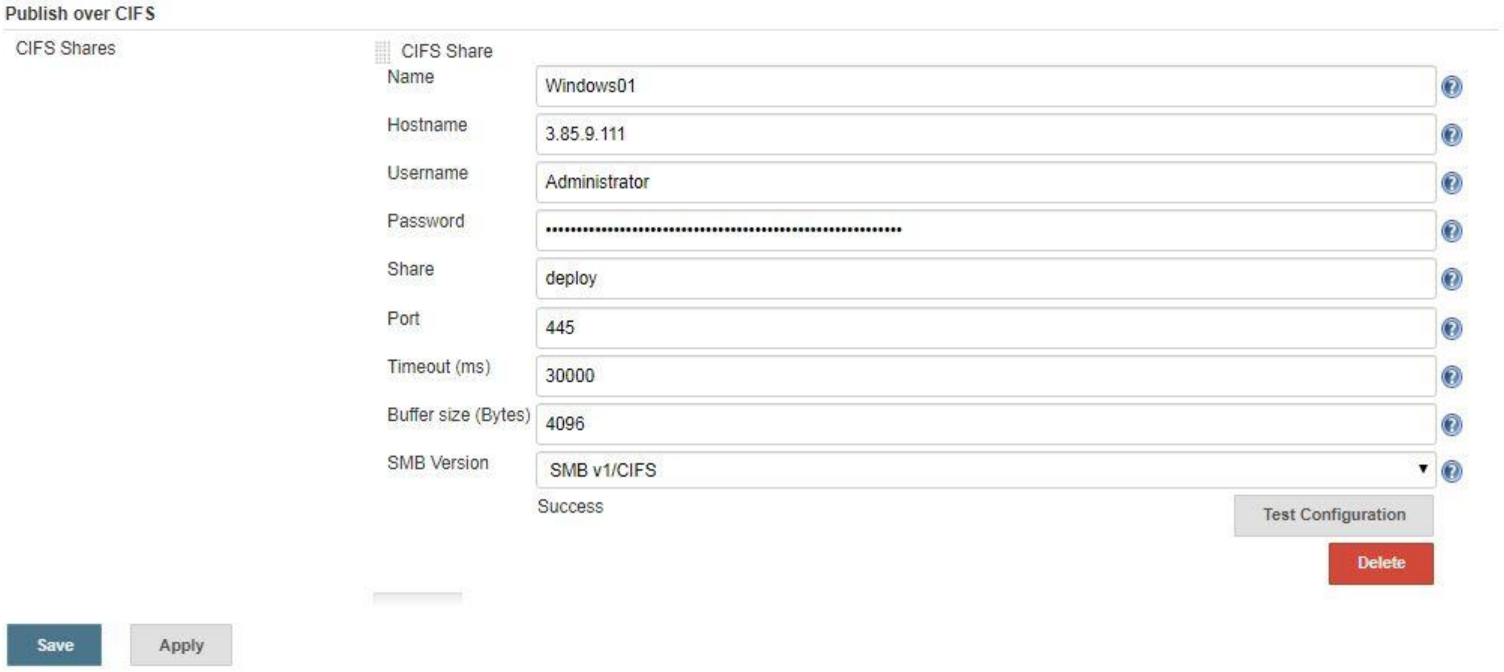
Manage Jenkins

**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 CIFS**


Publish over CIFS

CIFS Shares

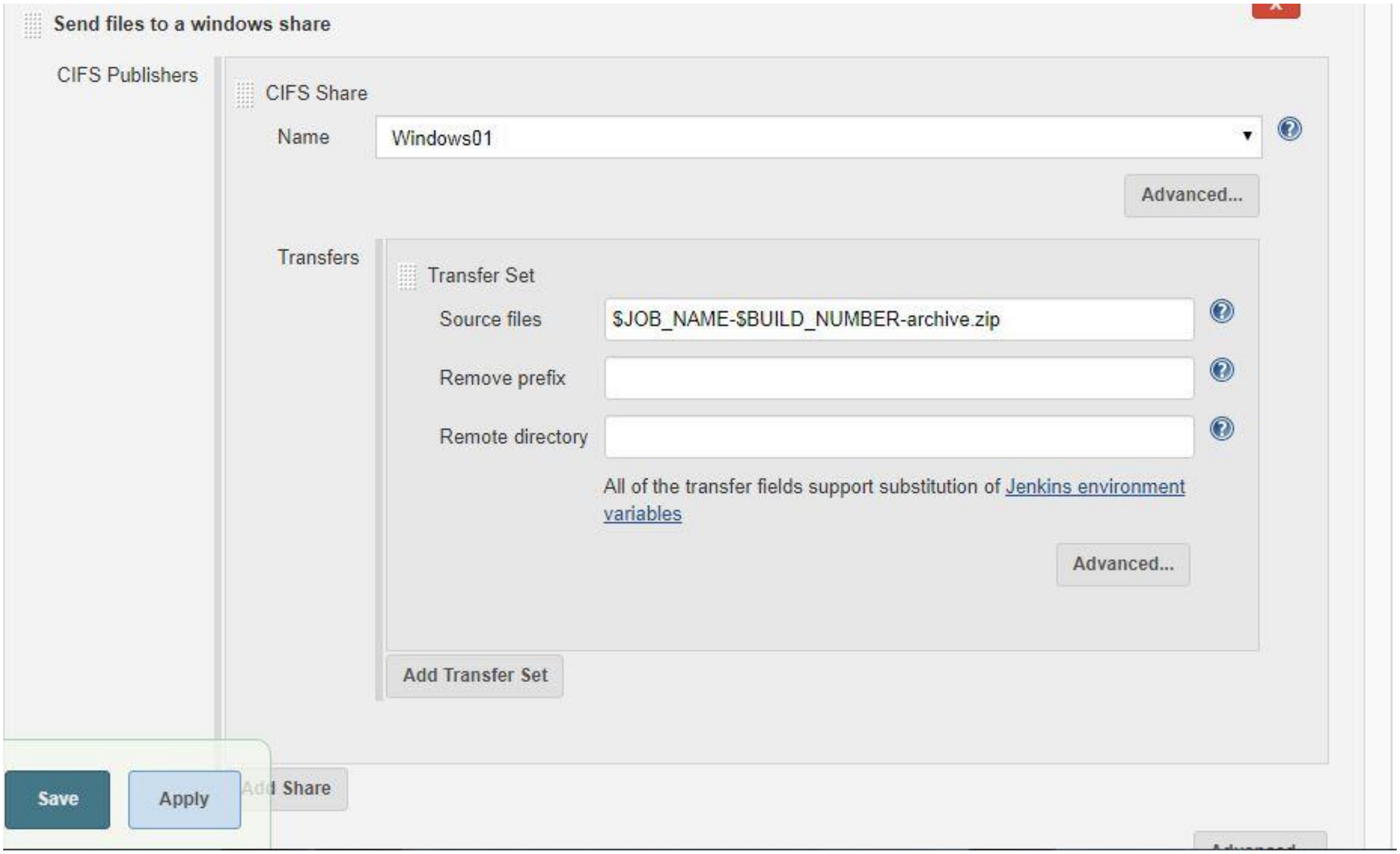
CIFS Share

| | |
|---|---|
| Name | Windows01 |
| Hostname | 3.85.9.111 |
| Username | Administrator |
| Password | •••••••••••••••••••••••••••••••••••••••••••••••••••••••••• |
| Share | deploy |
| Port | 445 |
| Timeout (ms) | 30000 |
| Buffer size (Bytes) | 4096 |
| SMB Version | SMB v1/CIFS |

Success

Provide username Password, Open Port 445 and create the folder deploy and share in n/w.

**Step-8**: Once you added the server in configure systems then return to your Build plan and now in Build options select “**Send files to windows share**”. You will see the host added there in scroll down list, select your server.



Send files to a windows share

CIFS Publishers

CIFS Share

Name: Windows01

Transfers

Transfer Set

Source files: $JOB_NAME-$BUILD_NUMBER-archive.zip

Remove prefix:

Remote directory:

All of the transfer fields support substitution of Jenkins environment variables

**Step-9**: Go to windows server (web server)

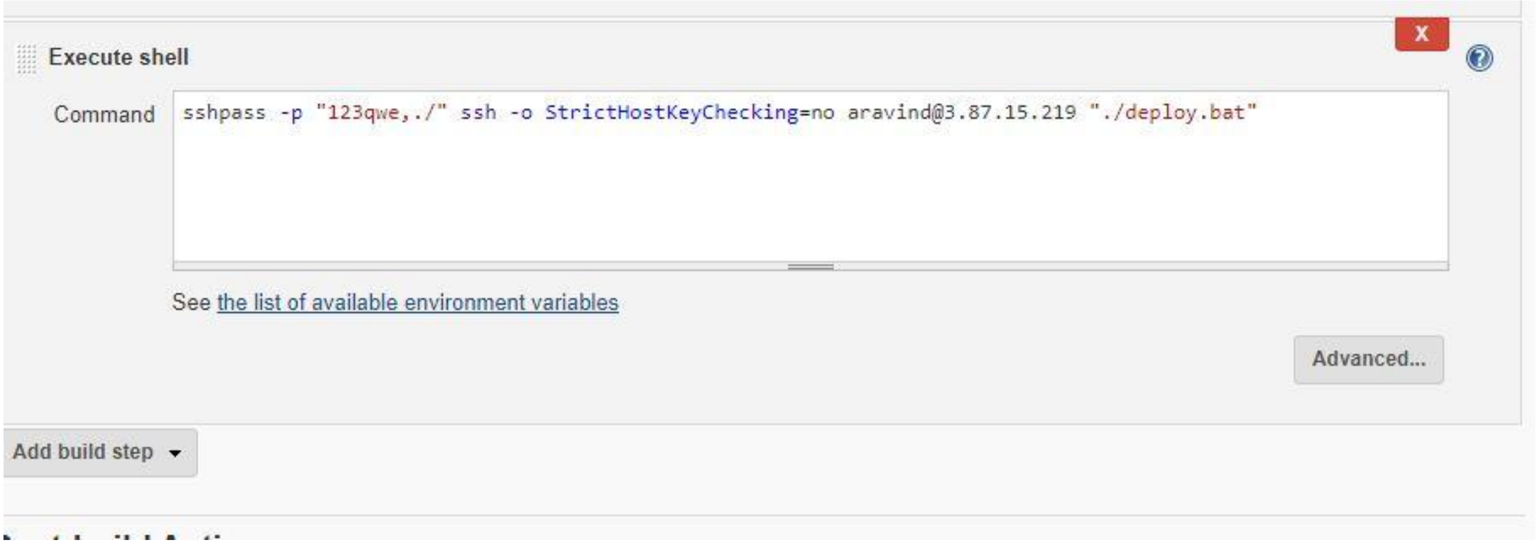
Install 7 zip and open ssh to invoke the script from Jenkins server.

Go to the location c:\users\<username>\ and create build.bat file and use the below deployment steps in script.

```
set PATH=C:\Program
Files\OpenSSH\bin;C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\Win
dowsPowerShell\v1.0;C:\Program Files\Amazon\cfn-bootstrap;C:\Program Files\OpenSSH\bin;C:\Program Files\7-
Zip
cd C: \deploy\
 7z e * -y
cd C:\inetpub\wwwroot\
rm -rf *
cd C:\ deploy\
xcopy * C:\inetpub\wwwroot\ /Y /S
```

**Step 10**: Invoke the bat script from Jenkins using open ssh command.

```
sshpass -p "<password>" ssh -o StrictHostKeyChecking=no username@<ipaddress> "./build.bat"
```



Execute shell

Command

```
sshpass -p "123qwe,./" ssh -o StrictHostKeyChecking=no aravind@3.87.15.219 "./deploy.bat"
```

See the list of available environment variables

Run the build option and check the deployment.