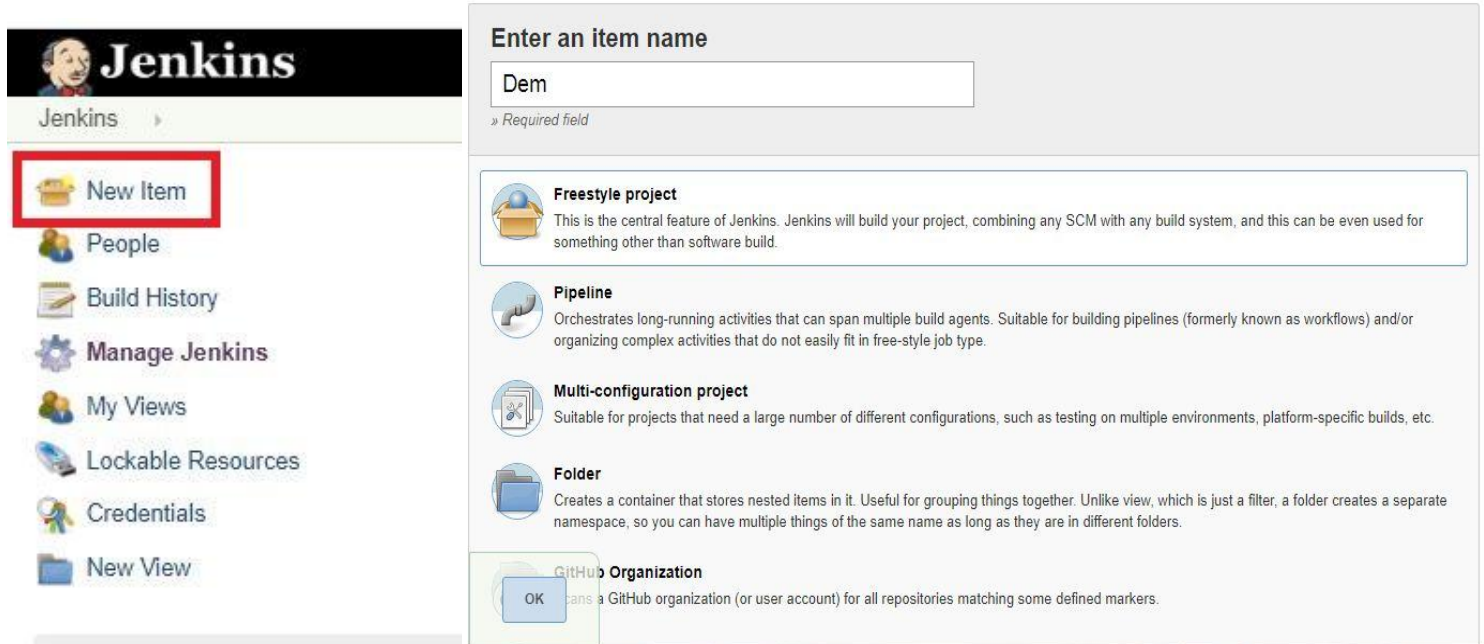
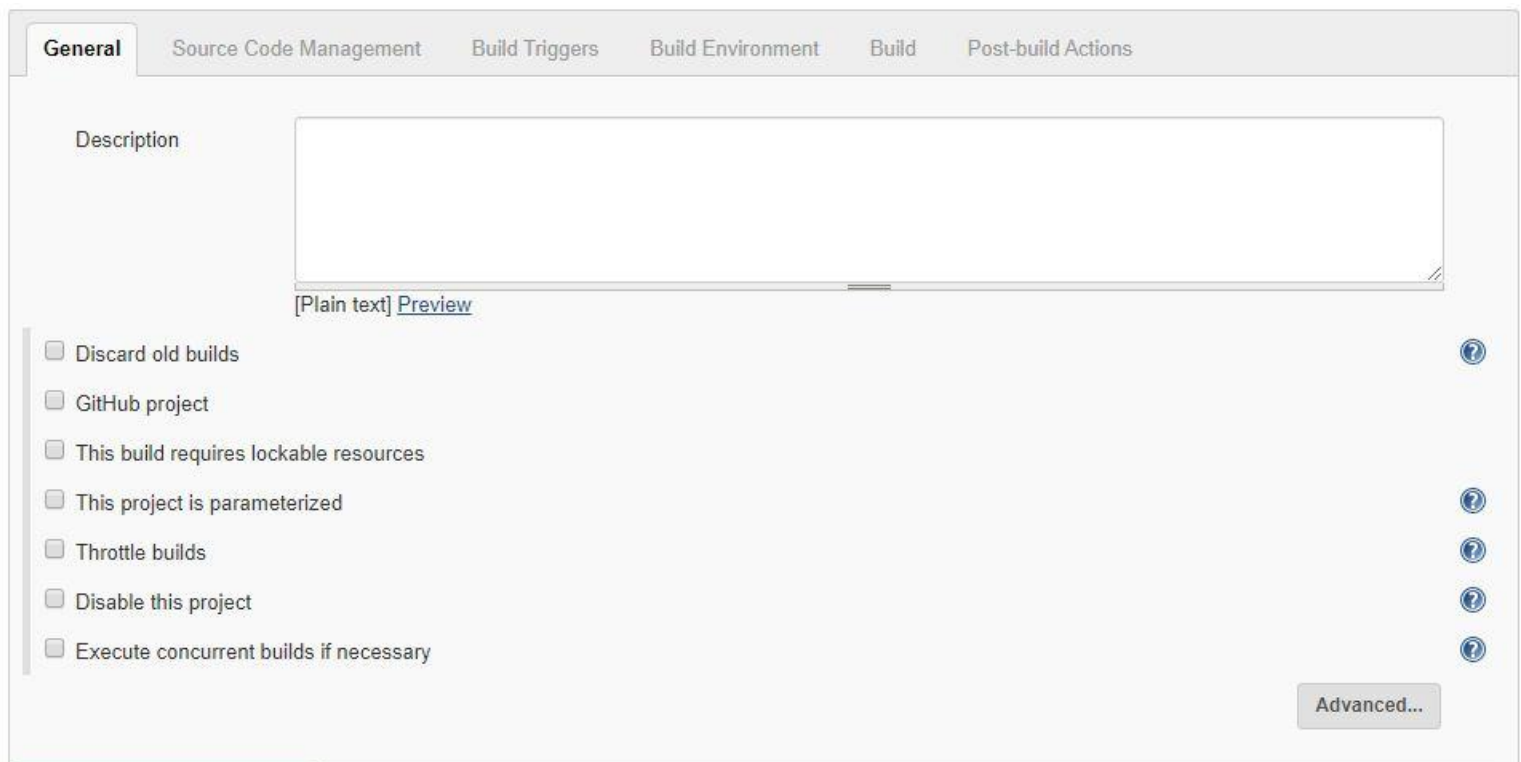


**Step1:** Click 'new item' in the dashboard. It will redirect to wizard provide 'item name' and type of project select 'Freestyle Project'.



The screenshot shows the Jenkins dashboard on the left with a sidebar containing links like 'New Item', 'People', 'Build History', 'Manage Jenkins', 'My Views', 'Lockable Resources', 'Credentials', and 'New View'. The 'New Item' button is highlighted with a red rectangle. On the right, the 'Enter an item name' wizard is open, showing a text input field with 'Dem' and a 'Required field' message. Below this, several project types are listed: 'Freestyle project' (selected), 'Pipeline', 'Multi-configuration project', 'Folder', and 'GitHub Organization'. Each option has a brief description. An 'OK' button is visible at the bottom of the wizard.

**Step2:** Once the freestyle project wizard open provide the description and check the option which is needed.



The screenshot shows the 'General' tab of the Jenkins project configuration wizard. At the top, there are tabs for 'General', 'Source Code Management', 'Build Triggers', 'Build Environment', 'Build', and 'Post-build Actions'. The 'General' tab is active, showing a 'Description' field with a placeholder '[Plain text] Preview'. Below the description field, there are several checkboxes: 'Discard old builds', 'GitHub project', 'This build requires lockable resources', 'This project is parameterized', 'Throttle builds', 'Disable this project', and 'Execute concurrent builds if necessary'. On the right side of the checkboxes, there are help icons (question marks). At the bottom right, there is an 'Advanced...' button.



**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](#)

Advanced...

## 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



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

**Serverin/ 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**

Advanced...

Test Configuration

Delete

Add

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

Test Configuration



**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.

The screenshot shows the Jenkins configuration interface for 'Send files or execute commands over SSH'. The window has a title bar with a red close button and a help icon. On the left, there's a sidebar with 'SSH Publishers' and 'SSH Server' sections. The 'SSH Server' section has a 'Name' dropdown menu set to 'Web01' and an 'Advanced...' button. The 'Transfers' section is expanded, showing a 'Transfer Set' with four fields: 'Source files' (containing '\$JOB\_NAME-\$BUILD\_NUMBER-archive.zip'), 'Remove prefix' (empty), 'Remote directory' (empty), and 'Exec command' (containing a multi-line shell script). Below the 'Exec command' field, there's a note: 'All of the transfer fields (except for Exec timeout) support substitution of [Jenkins environment variables](#)'. At the bottom left, there are 'Save' and 'Apply' buttons. At the bottom right, there's another 'Advanced...' button.

Send files or execute commands over SSH

SSH Publishers

SSH Server

Name Web01

Advanced...

Transfers

Transfer Set

Source files \$JOB\_NAME-\$BUILD\_NUMBER-archive.zip

Remove prefix

Remote directory

Exec command

```
cd /var/www
sudo rm -rf dcc
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
```

All of the transfer fields (except for Exec timeout) support substitution of [Jenkins environment variables](#)

Advanced...

Save Apply

```
cd /var/www/
sudo rm -rf dcc
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-
$BUILD_NUMBER
cd /var/www/
sudo ln -s deployments/$JOB_NAME-$BUILD_NUMBER dcc
```

The above script will extract the source artifacts from **/deploy** to **/deployments/\$JOB\_NAME-\$BUILD\_NUMBER** and then create simlink (softlink) **dcc** which pointing to **/deployments/\$JOB\_NAME-\$BUILD\_NUMBER** so that current version will be loaded in site.

This plan based on PHP deployment try the same with your own options.