

Connecting Jenkins with Backend instance and creating freestyle job and CI/CD Pipeline

Step 1: Generate RSA Key Pair on Jenkins Server

Connect to your Jenkins EC2 instance via SSH.

```
ssh -i path_to_your_key.pem ec2-user@your_ec2_public_dns
```

Switch user to jenkins

```
sudo su - jenkins
```

```
ubuntu@ip-172-31-40-129:~$ sudo su - jenkins
jenkins@ip-172-31-40-129:~$ ls -l
total 76
-rw-r--r-- 1 jenkins jenkins 1660 Oct 17 08:15 config.xml
-rw-r--r-- 1 jenkins jenkins 156 Oct 17 08:15 hudson.model.UpdateCenter.xml
-rw-r--r-- 1 jenkins jenkins 370 Oct 17 08:22 hudson.plugins.git.GitTool.xml
-rw----- 1 jenkins jenkins 1680 Oct 17 08:22 identity.key.enc
-rw-r--r-- 1 jenkins jenkins 7 Oct 17 08:24 jenkins.install.InstallUtil.lastExecVersion
-rw-r--r-- 1 jenkins jenkins 7 Oct 17 08:24 jenkins.install.UpgradeWizard.state
```

Generate the RSA key pair:

```
ssh-keygen -t rsa -b 2048
```

```
jenkins@ip-172-31-40-129:~$ ssh-keygen -t rsa -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key (/var/lib/jenkins/.ssh/id_rsa):
Created directory '/var/lib/jenkins/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/lib/jenkins/.ssh/id_rsa
Your public key has been saved in /var/lib/jenkins/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Bt/ZuMxQKfqCRGHwt9pJxN94xAF7X6moXN+z/11gGKE jenkins@ip-172-31-40-129
The key's randomart image is:
+---[RSA 2048]---+
| ..0   . . . |
| o o   o + . . |
|   o = o E . o |
| . o * O * =   |
| . + S O = o   |
| . = B o o .   |
| o + + + . o . |
|           oo|
|           ..=|
+---[SHA256]-----+
```

Step 2: Copy the Public Key to the Backend Server

Get the public key:

```
cat ~/.ssh/id_rsa.pub
```

```
jenkins@ip-172-31-40-129:~$ cd .ssh/  
jenkins@ip-172-31-40-129:~/.ssh$ ls -l  
total 8  
-rw----- 1 jenkins jenkins 1831 Oct 17 09:21 id_rsa  
-rw-r--r-- 1 jenkins jenkins 406 Oct 17 09:21 id_rsa.pub
```

```
jenkins@ip-172-31-12-13:~/.ssh$ cat id_rsa.pub  
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDlmlXg1hrGDG3SVW65hh8YbM2eIC3h0S7vgWQJvth6p9z16v18AvkiJehd  
+rTFeWCocdhz1y9UbI/HSVTaqoahbKuB1kwbQqf1UEWJAb8THqed/C1zKxR8UglZSgc5bid8qhh3LiAGiFgdi6Mp+xSC1JE4q  
+JuVVHrb3U4FAOKIawQpuMLSGpjf3LMebdrsIJfELoKjRoXE1Ev02W9LENFtmTw7n4otQ2Iqc4jm90hxzFx0GfwevyQQ41vS
```

Connect to your backend server instance via SSH and Add the public key to the **authorized_keys**:

```
~/.ssh/authorized_keys
```

```
nano ~/.ssh/authorized_keys
```

Paste the public key you copied from the Jenkins instance. and Save the file and exit.

Step 3: Test SSH Connection from Jenkins

Back on the Jenkins instance, test the SSH connection to the backend server:

```
ssh ubuntu@backend-server-private-ip
```

```
jenkins@ip-172-31-12-13:~/.ssh$ ssh ubuntu@172.31.6.72  
The authenticity of host '172.31.6.72 (172.31.6.72)' can't be established.  
ED25519 key fingerprint is SHA256:EgN3h+MC3vTMuzVfE0LOZwBzwY4w+BOUdVI2SsCJo5s.  
This key is not known by any other names  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added '172.31.6.72' (ED25519) to the list of known hosts.  
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1015-aws x86_64)
```

```
*** System restart required ***
```

```
Last login: Thu Oct 17 12:09:34 2024 from 115.244.184.254  
ubuntu@ip-172-31-6-72:~$ ls  
fundoo-notes  
ubuntu@ip-172-31-6-72:~$ exit  
logout  
Connection to 172.31.6.72 closed.
```

Create a New Freestyle Project in Jenkins



- From the Jenkins dashboard, click on New Item.
- Enter a name for your job (e.g., `DjangoAppBuild`) and Select Freestyle project.

Dashboard > All > New Item

New Item

Enter an item name

Select an item type

 **Freestyle project** 
Classic, general-purpose job type that checks out from up to one SCM, build steps like archiving artifacts and sending email notifications.

Configure Project

- In the job configuration page, scroll to General

Dashboard > demo-first-job >

 **Status**


 **Changes**


 **Workspace**


 **Build Now**


 **Configure** 


Configure


 General

 Source Code Management

 Build Triggers

 Build Environment

 Build Steps

 Post-build Actions

Description

first project demo

Plain text [Preview](#)

☒ Discard old builds ?

Strategy

Log Rotation

Days to keep builds

if not empty, build records are only kept for

5


Max # of builds to keep


if not empty, only up to this number of


Configure Git repository


- In the job configuration page, scroll down to Source Code Management.
- Select Git.
- Enter your repository URL (e.g., `git@github.com:username/repo.git`).
- If needed, configure credentials by clicking on Add next to Credentials.


Configure


 General

 Source Code Management

 Build Triggers

 Build Environment

 Build Steps

 Post-build Actions

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/ayush-prajapati01/fundoo-notes-copy.git

Credentials ?

- none -

Configure Build Steps

Here we configure our build

Select Discard Old builds

Build Environment

- ☒ Delete workspace before build starts
- ☐ Use secret text(s) or file(s) ?
- ☐ Add timestamps to the Console Output
- ☐ Inspect build log for published build scans
- ☐ Terminate a build if it's stuck
- ☐ With Ant ?

Build Steps

```
set -e # Exit immediately if a command exits with a non-zero status

echo "Syncing the application to backend/app server"
rsync -avz $WORKSPACE ubuntu@172.31.6.72:/tmp || { echo "rsync failed"; exit 1; }

ssh ubuntu@172.31.6.72 'sudo cp -rfv /tmp/fundoo-automate/*
/home/ubuntu/fundoo-notes/' || { echo "Copying files failed"; exit 1; }

ssh ubuntu@172.31.6.72 'source /home/ubuntu/fundoo-notes/venv/bin/activate &&
pip3 install -r /home/ubuntu/fundoo-notes/requirements.txt && pip3 install django' ||
{ echo "Installation failed"; exit 1; }

ssh ubuntu@172.31.6.72 'source /home/ubuntu/fundoo-notes/venv/bin/activate &&
sudo python3 /home/ubuntu/fundoo-notes/fundoo_notes/manage.py migrate' || {
echo "Migration failed"; exit 1; }

echo "Restarting Django service"
ssh ubuntu@172.31.6.72 'sudo systemctl restart fundooauto.service && sudo
systemctl status fundooauto.service' || { echo "Failed to restart the service"; exit 1;
}
```

Save and Build

- Click on the Save button at the bottom of the configuration page.

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

```
ssh ubuntu@172.31.1.175
ssh ubuntu@172.31.1.175
```

Advanced ▾

Add build step ▾

Post-build Actions

Add post-build action ▾

Save

Apply

- To run your job, go back to the job page and click on Build Now.

</> Changes

Workspace

▶ Build Now

Configure

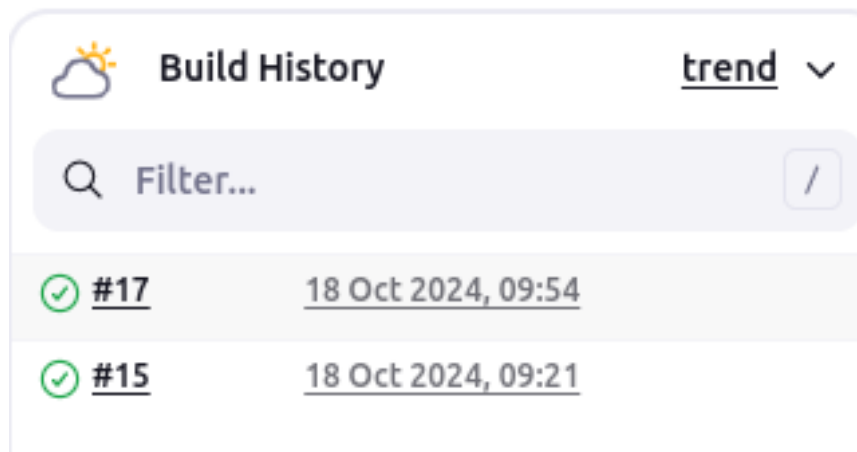
Delete Project

first project demo

Permalinks

- [Last build \(#17\), 2 days 20 hr ago](#)
- [Last stable build \(#17\), 2 days 20 hr ago](#)
- [Last successful build \(#17\), 2 days 20 hr ago](#)
- [Last completed build \(#17\), 2 days 20 hr ago](#)

- Monitor the build process by clicking on the build number in the build history.



Create a New Pipeline Project in Jenkins

Create New Job

- From the Jenkins dashboard, click on New Item.
- Enter a name for your job (e.g., DjangoAppPipeline).
- Select Pipeline and click OK.

Configure Pipeline Script

- In the job configuration page, scroll down to the Pipeline section.
- Choose Pipeline script from the Definition dropdown.

Configure

⚙️ General

🔑 Advanced Project Options

📄 Pipeline

Pipeline

Definition

Pipeline script

Pipeline script

Pipeline script from SCM

Enter Pipeline Script

Here's the basic pipeline script for our Django application

```

pipeline {
    agent any

    stages {
        stage('Checkout Code') {
            steps {
                // Checkout the latest code from the specified Git repository
                git branch: 'dev', url: 'https://github.com/Deven5656/fundoo-notes.git'
            }
        }

        stage('Sync Application') {
            steps {
                script {
                    echo "Syncing the application to backend/app server"
                    def syncStatus = sh(script: "rsync -avz $WORKSPACE
ubuntu@172.31.6.72:/tmp", returnStatus: true)
                    if (syncStatus != 0) {
                        error "rsync failed"
                    }
                }
            }
        }

        stage('Copy Files') {
            steps {
                script {
                    echo "Copying files to the server"
                    def copyStatus = sh(script: "ssh ubuntu@172.31.6.72 'sudo cp -rfv
/tmp/fundoo-pipeline/* /home/ubuntu/fundoo-notes/'", returnStatus: true)
                    if (copyStatus != 0) {
                        error "Copying files failed"
                    }
                }
            }
        }

        stage('Install Dependencies') {
            steps {
                script {
                    echo "Installing dependencies"
                    def installStatus = sh(script: "ssh ubuntu@172.31.6.72 'source
/home/ubuntu/fundoo-notes/venv/bin/activate && pip3 install -r
/home/ubuntu/fundoo-notes/requirements.txt && pip3 install django'", returnStatus:
true)
                    if (installStatus != 0) {
                        error "Installation failed"
                    }
                }
            }
        }
    }
}

```



```

    }
  }
}

stage('Run Migrations') {
  steps {
    script {
      echo "Running migrations"
      def migrateStatus = sh(script: "ssh ubuntu@172.31.6.72 'source
/home/ubuntu/fundoo-notes/venv/bin/activate && sudo python3
/home/ubuntu/fundoo-notes/fundoo_notes/manage.py migrate'", returnStatus: true)
      if (migrateStatus != 0) {
        error "Migration failed"
      }
    }
  }
}

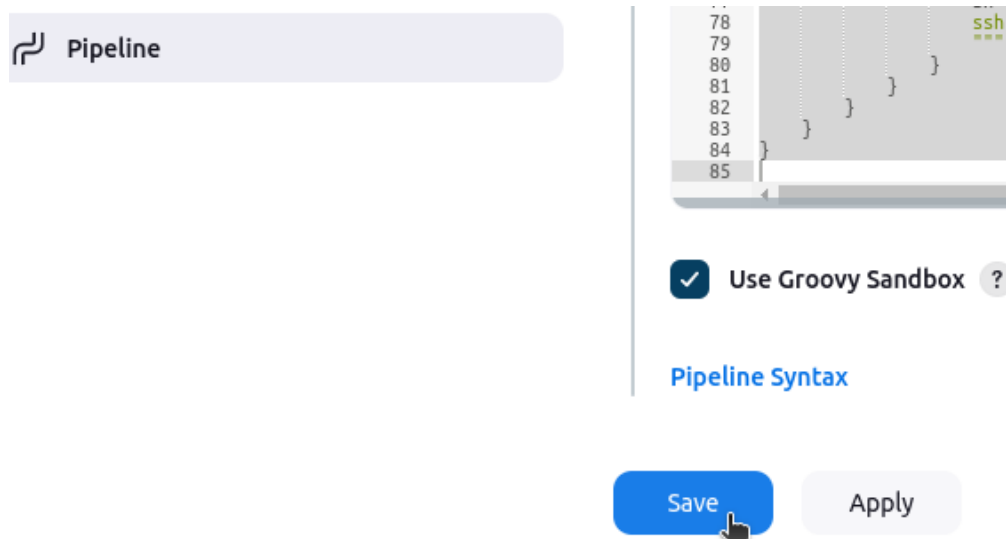
stage('Restart Django Service') {
  steps {
    script {
      echo "Restarting Django service"
      def restartStatus = sh(script: "ssh ubuntu@172.31.6.72 'sudo systemctl
restart fundooauto.service && sudo systemctl status fundooauto.service'",
returnStatus: true)
      if (restartStatus != 0) {
        error "Failed to restart the service"
      }
    }
  }
}

post {
  failure {
    echo "Pipeline failed!"
  }
  success {
    echo "Pipeline completed successfully!"
  }
}
}

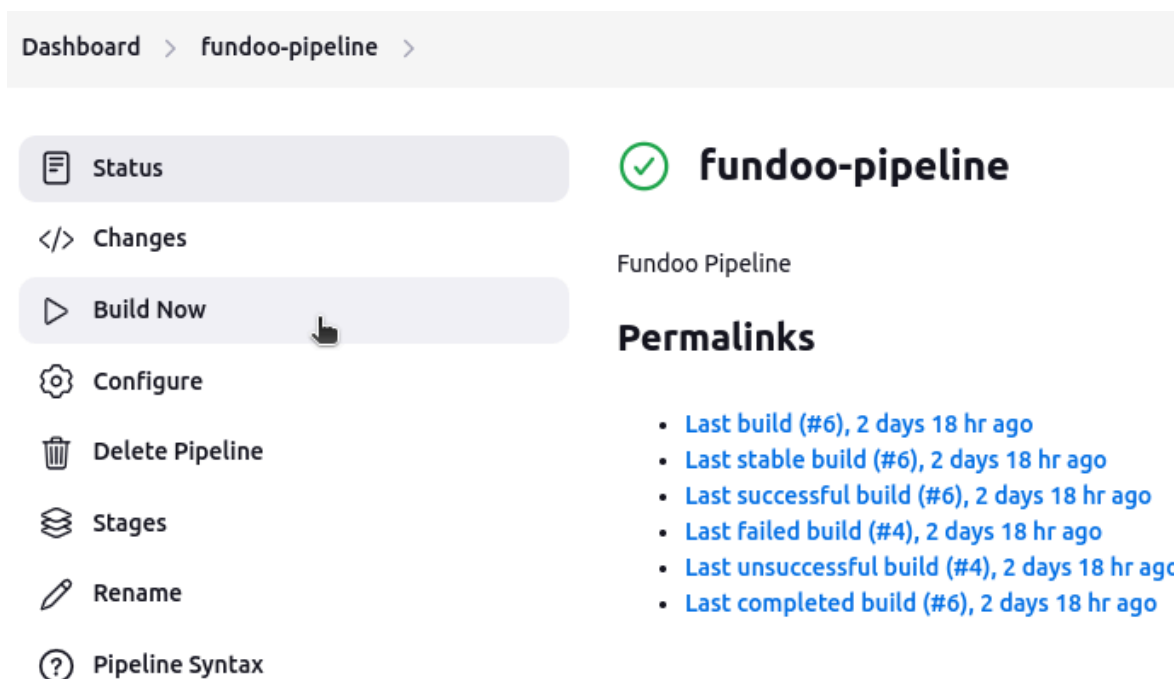
```

Save and Build

1. Click on the Save button at the bottom of the configuration page.



2. To run your job, go back to the job page and click on Build Now.



3. Monitor the build process by clicking on the build number in the build history.

