

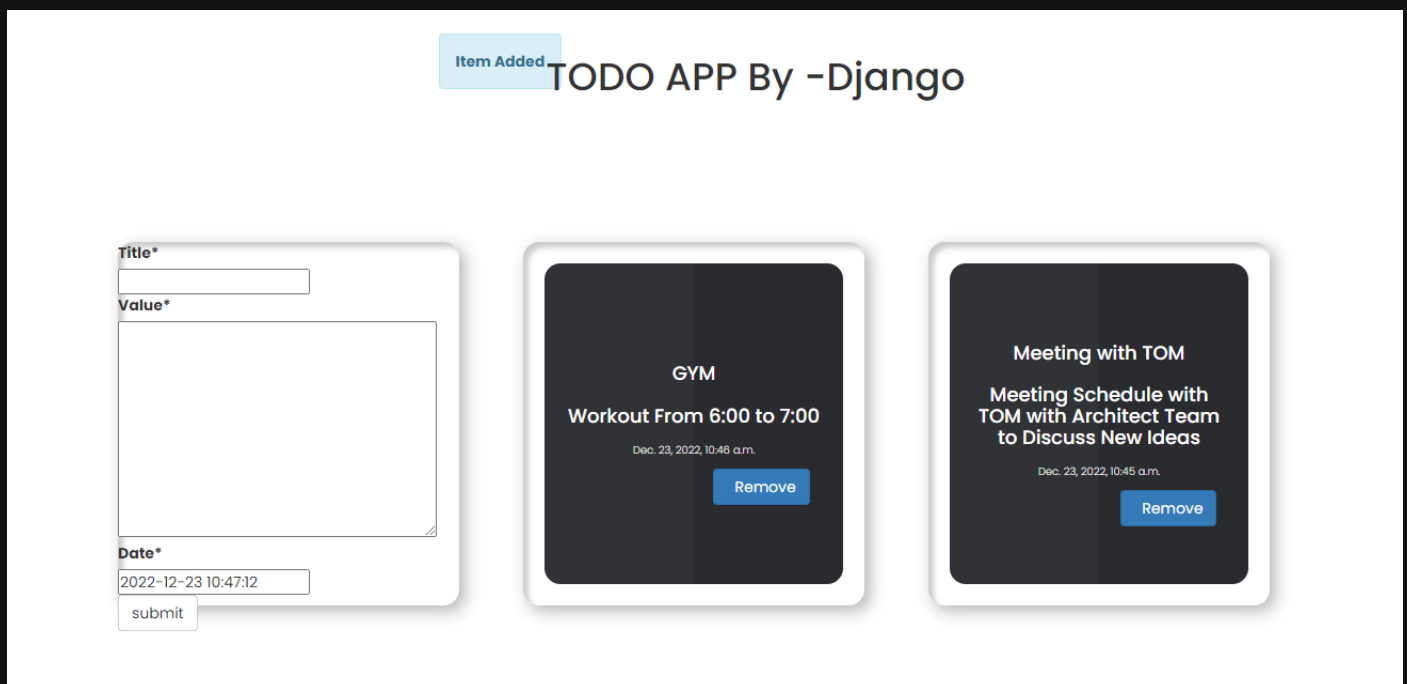
Project: Deploy Docker Provision Django Application on AWS with Default server & Include CI/CD with Jenkins.

Pre-requisites:

1. AWS Account
2. GitHub Account
3. Basic Knowledge About Docker & Jenkins

Steps:

i) Clone the Django TODO App



- ii) Launch an EC2 Ubuntu Instance with access of HTTP & HTTPS
- iii) Install Jenkins on Ubuntu Machine
- iv) Setup Jenkins & connect with GIT
- v) Finally Enable the Jenkins Script for CI/CD of Docker Provision Django Application

Step 1:

Clone the Django TODO app by running Below Command & push that Code to Your Personal Repository

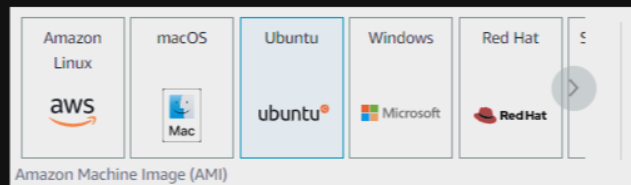
```
git clone https://github.com/tushargangurde2029/django-todo.git
```

Repository to use

Note:- You need to push this application to your Personal Repository so you can add ssh keys & add Webhooks to implement CI/CD Integration

Step 2:

Go to your AWS Account & Launch an Ubuntu EC2 Instance
Open EC2 -> Instances -> Launch an EC2 Instance
Select Ubuntu Image



Provide Following Details

Name = SampleDjangoApp

Instance Type = t2.micro

Allow HTTP & HTTPS Traffic From Internet


Once Done Launch Your EC2 Instance

Note: Please create a new key pair or use existing one to login

Step 3:


Connect to your EC2 Instance to Install Jenkins

RUN Below Commands One by One
Update Your System




```
sudo apt update
```

Install JAVA



```
sudo apt install openjdk-11-jre
```

Check JAVA Version by running below command



```
java -version
```

Now Install Jenkins



```
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \ https://pkg.jenkins.io/debian binary/  
| sudo tee \ /etc/apt/sources.list.d/jenkins.list > /dev/null
```



```
curl -fsSL https://pkg.jenkins.io/debian/jenkins.io.key | sudo tee \ /usr/share/keyrings/jenkins-  
keyring.asc > /dev/null
```



```
sudo apt-get update
```



```
sudo apt-get install jenkins
```

Once Done Enable & start the Jenkins Service



```
sudo systemctl enable jenkins
```



```
sudo systemctl start jenkins
```

Check Jenkins is successfully started by below command

```
sudo systemctl status jenkins
```

Now GO-TO Security Group of your EC2 Instance & Provide below Inbound Rules & Save Changes

Custom TCP	TCP	8080	Anywh...	0.0.0.0/0	Delete
Custom TCP	TCP	8000	Anywh...	0.0.0.0/0	Delete

To Access Jenkins Server

To Access Django Application

Now Open your Jenkins Server by Below address

http://[public-ip]:8080/

You can see Below Screen

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

Continue

Step 4:

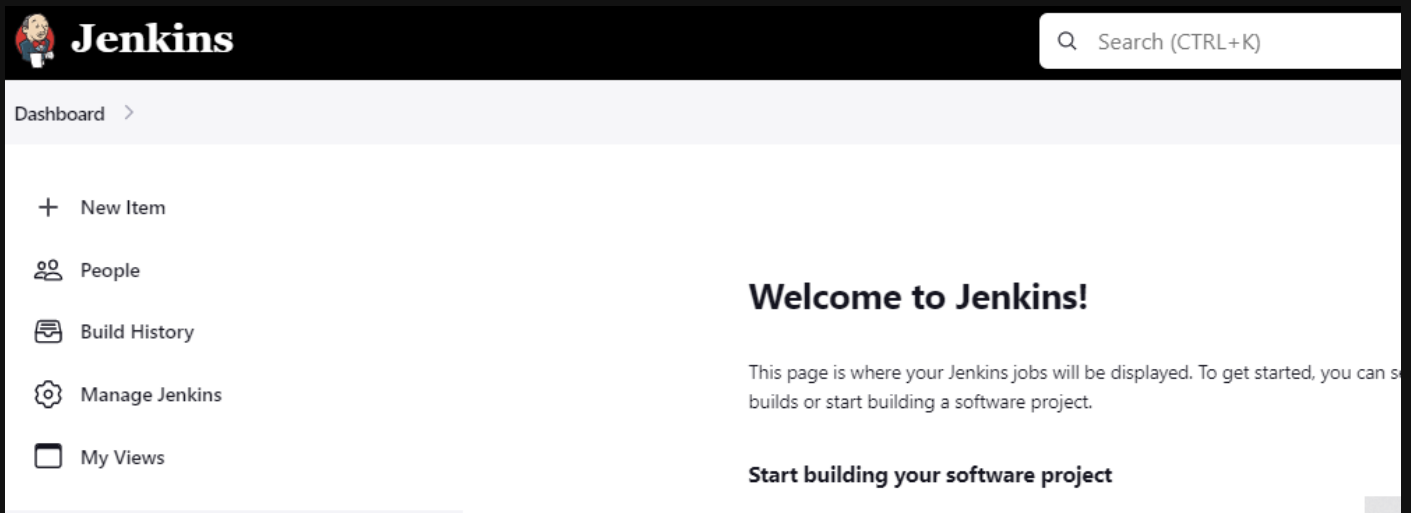
Now locate your Jenkins administrator password by command

```
ubuntu@ip-172-31-34-232:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
e88fedcbc52a41dfb93f756d5c99d571
```

Enter that Password & select Install Suggested Plugins
Once Done Provide the Necessary Details & click on
Save & Continue

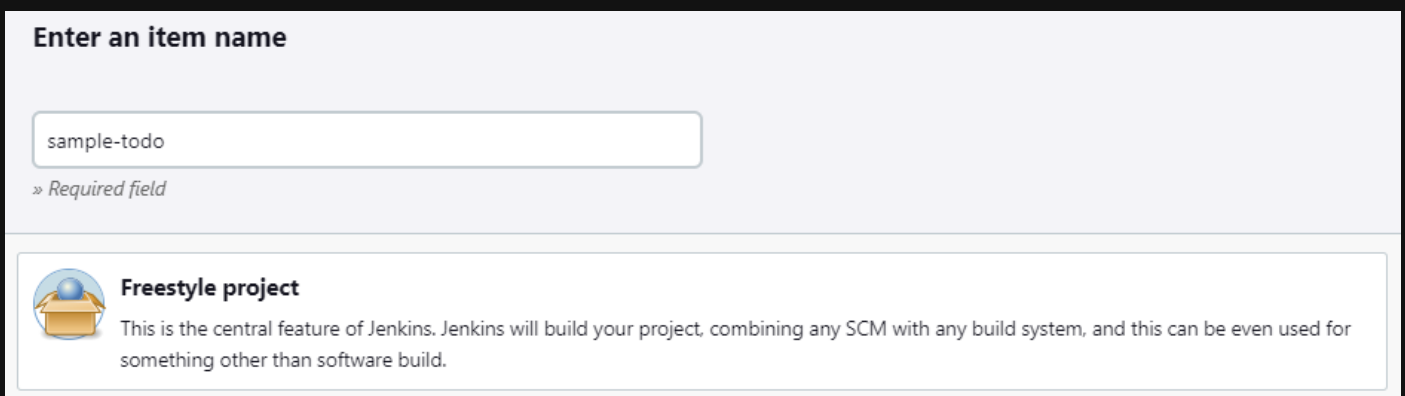
Check the Jenkins URL & click on Save & Finish

Now click on Start Using Jenkins You can see below screen



The screenshot shows the Jenkins dashboard. At the top left is the Jenkins logo and name. To the right is a search bar with the text "Search (CTRL+K)". Below the logo is a sidebar with a "Dashboard" link and a list of menu items: "New Item", "People", "Build History", "Manage Jenkins", and "My Views". The main content area has a heading "Welcome to Jenkins!" followed by a paragraph: "This page is where your Jenkins jobs will be displayed. To get started, you can s builds or start building a software project." At the bottom of the main area is a button that says "Start building your software project".

Provide Item Name, we are using freestyle pipeline so
choose freestyle project



The screenshot shows the "Enter an item name" form in Jenkins. It has a text input field containing "sample-todo". Below the field is a small text label "» Required field". Below the form is a section titled "Freestyle project" with a blue circular icon containing a white box. To the right of the icon is a paragraph: "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."

Once Done click on Save

Before configuring lets connect Jenkins with git using SSH

GO to your EC2 Instance & run below command

```
ubuntu@ip-172-31-34-232:~$ ssh-keygen
```

It will generate public & private key provide public key for GitHub & provide private key for Jenkins
Access the keys by changing the directory to

```
ubuntu@ip-172-31-34-232:~$ cd .ssh
ubuntu@ip-172-31-34-232:~/.ssh$ ls
authorized_keys  id_rsa  id_rsa.pub
```

Now GO to your GitHub & provide the public key as

```
ubuntu@ip-172-31-34-232:~/.ssh$ cat id_rsa.pub
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQgQDC7t+X5gu9NK/hKaFV0UXaa7eHHZSsXR4uSEpkeP6vjt+MnDDhXnc67BxtT08/2tQFf0paPjm
7W+bMgxUJJMbpTfFQfjMEgGXHJ8sjvkZQRtL8i1ScXCLWNpV/LyCfzxqEg/zo0plVttbZi7cTCJjNLi6snW2xq5IbGBikCu0e4kbvFC
aXJ4mCaHB4NLWkw35qz+wmWfaDPLlYyYXijVfxZ7LV0IVm1xGelFo42wmvaImiY6/oJP1didiVsrD0o1CNt5AtzjMQ0RVQkKh9pRab
HF2E+yw7ZS0jERXS0tiiFw1FC/MyVpLzGZwDoB8mR65VRxrKUEIzfp6JBZJmd/+EjqGa28kSfeKld3NKeHL7iEUmcAex0C3Cg3h4nUs
NVYyu0UsYgtqa0arRZdqmld8gMJ2IwFL+R2z5vTQFzZ0TMJJCUxxZAn9Ga4xvTYFgSXLqt05iu03jSH7Gr/MmGlbowZsFSBkGLGH2Pt
4jDJCoTGjgUUIGp4cEa0w+XY8V2k= ubuntu@ip-172-31-34-232
```

SSH keys / Add new

Title

Sample

Key type

Authentication Key

Key

ssh-rsa

```
AAAAB3NzaC1yc2EAAAADAQABAAQGBgQDC7t+X5gu9NK/hKaFV0UXaa7eHHZSsXR4uSEpkeP6vjt+MnDDhXnc67BxtT08/  
2tQFf0paPjm7W+bMgxUJJMbpTfFQfjMEgGXHJ8sjvkZQRtL8i1ScXCLWNpV/LyCfzxqEg/zoOplVttbZi7cTCJjNLI6snW2xq5Ib  
GBikCu0e4kbvFCaXJ4mCaHB4NLWkw35qz+wmWfaDPLlYyYXYijVfxZ7LVOIVm1xGelFo42wmvalmiY6/oJP1didiVsrDOo1CN  
t5AtzjMQ0RVQkKh9pRabHF2E+yW7ZS0jERXS0tiiFw1FC/MyVpLzGZwDoB8mR65VRxrKUElzf6JBZJmd/+EjqGa28kSfeKId  
3NKeHL7iEUmcAexOC3Cg3h4nUsNVYyuOUsYgtqa0arRZdqmID8gMJ2lwFL+R2z5vTQFzZOTMJJCuxxZAn9Ga4xvTYFgSXL  
qtO5iu03jSH7Gr/MmGlbwoZsFSBkGLgH2Pt4jDJCoTGjgUUIGp4cEa0w+XY8V2k= ubuntu@ip-172-31-34-232
```

Add SSH key

As you can see i have added the SSH key for the GitHub

Now Similarly add private key to your Jenkins
GO to your Project --> Configure
Check GitHub Project



GitHub project

Project url ?

https://github.com/tushargangurde2029/django-todo.git

Advanced...

In Source Code Management select GIT and paste the repository URL

Git ?

Repositories ?

Repository URL ?

`https://github.com/tushargangurde2029/django-todo.git`

Credentials ?

- none -

+ Add

Now in Credentials Click on add

Provide the Details

In Kind select SSH with Private Key

Username select as Ubuntu --> Username of EC2 Instance

In Private Key select Entire Directly & paste your Private Key as we copied public key

```
ubuntu@ip-172-31-34-232:~/.ssh$ cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAABG5vbmUAAAABbm9uZQAAAAAAAAABAAABlwAAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAwu7fl+YLvTSv4SmhVdFF2mu3hx2UrF0eLkhKZJ+r47fjJww4V53
0uwcB09PP9rUBX9KWj45u1vmzIMVCSTG6U3xUH4zBIBlxyfLI75GUEbS/ItUnFwi1jaVfy
8gn88ahIP86DqZVbbW2Yu3EwiYzS4urJltsauSGxgYpArtHuJG7xQmlyeJgmhweDSlpMN+
as/sJln2gzy5csmF2Io1X8Wey1TiFZtcRnpRa0NsJr2iJom0v6CT9XYnYlbKwzqNQjbeQL
c4zENEVUJCofaUWmxxdhPslu2UtIxEV0jrYohcNRQvzMlaS8xmcA6AfJkeuVUcaylBCM36
```

Enter directly

Key

```
H2IFpSCAYT7J0CILESMRF0I6ZKIXD00C8ZJWKVMZNA0GHyZHRIVHGSpQQJIN#HOKFRM23/45  
0oZrbyRJ94qV3c0p4cvuIRSZwB7E4LcKDeHidSw1VjK45SxiC2prRqtF12qaUPyAwnYjAU  
v5HbPm9NAXNk5MwkkJTHfKcF0ZrjG9NgWBjcuq07mK7TeNIfsav8yYaVvChmwVIGQYuAfY  
+3iMMkKhMa0BRQganhwRrTD5djxXaQAAAAMBAAEAAAAGAC2cA
```

Once Done Check Specifier for me it's main

Branches to build ?

Branch Specifier (blank for 'any') ?

*/main

Now Click on Save

After that click on Build Now

You can see build is started Once Done open that build
Go to console Output & copy the address



Console Output

Started by user [kratos](#)

Running as SYSTEM

Building in workspace /var/lib/jenkins/workspace/sample-todo

The recommended git tool is: NONE

Now open your Instance & change Directory with



```
ubuntu@ip-172-31-34-232:~/.ssh$ cd /var/lib/jenkins/workspace/sample-todo
ubuntu@ip-172-31-34-232:/var/lib/jenkins/workspace/sample-todo$ ls
Dockerfile  README.md  mainapp  manage.py  requirements.txt  todoZ
```

As you can see our project is present now lets install docker and build the docker image by following commands



```
ubuntu@ip-172-31-34-232: sudo apt install docker.io
```

Once Docker is installed build image by following command



```
ubuntu@ip-172-31-34-232: sudo docker build . -t todo
```

After successfully image is built run the image by



```
ubuntu@ip$ sudo docker run --name todo -d -p 8000:8000 todo
8751aae57f9b23e8f99642d182c8e02064e31873ff1c3d82ee56ccb067cad872
```

container name

port expose to run app

Image name

Verify the Application is running or not by below URL

[http:// \[public-ip\] :8000/](http://[public-ip]:8000/)

As you can see our application is running successfully

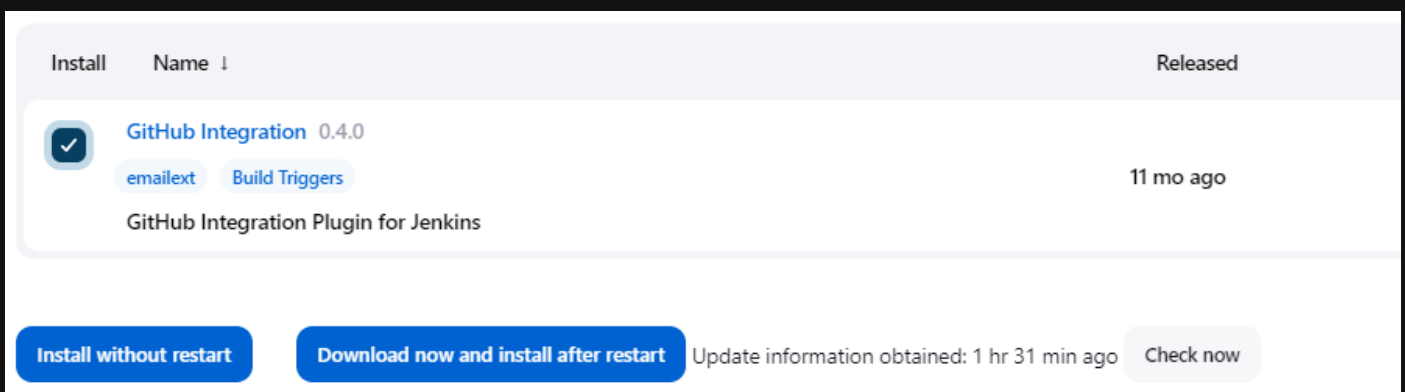


The screenshot shows a web application titled "TODO APP By -Django". It features a form with three fields: "Title*" (a text input), "Value*" (a large text area), and "Date*" (a date/time input showing "2022-12-23 13:15:28"). Below the form is a "submit" button.

Step 5:-

Let's add a git webhook to implement Continuous Delivery

First install GitHub Integration Plugin for Jenkins
Go to Jenkins --> Manage Jenkins --> Manage Plugins
In Available Plugins Search for GitHub Integration
Select the plugin & Install without Restart



The screenshot shows the Jenkins Manage Plugins interface. The "Available" tab is selected. A search for "GitHub Integration" has been performed, and the results show the "GitHub Integration" plugin (version 0.4.0) with a checkmark in the "Install" column. The "Released" column shows "11 mo ago". Below the plugin list, there are two buttons: "Install without restart" and "Download now and install after restart". The "Update information obtained: 1 hr 31 min ago" and a "Check now" button are also visible.

Once installation is done go to your Jenkins and configure
After opening Configuration check the dialog box as shown
in the image

Check the Dialog box in build triggers section

Once Done go to Build Steps section click on add build step
select execute shell

And add the below commands as shown in the image



After that click on Save

Now go to your GitHub open project repository click on setting

In Settings click on --> Webhook --> Add Webhook

Provide the Payload URL as

Jenkins-URL / github-webhook /

Refer the below Image for configurations

Payload URL *

http://3.6.39.66:8080/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.

After that click on Add Webhook

Once webhook is added go to your EC2 Instance give Jenkins access to run Docker commands by below command

```
sudo usermod -a -G docker jenkins
```

After that restart the jenkins server by below command

```
sudo systemctl restart jenkins
```

All the Steps are Successfully Completed.

Now try to change the code and push the changes to GIT
it will run a build and new docker image will be created &
it will be automatically going to deploy on your EC2 Instance



Console Output

```
Started by GitHub push by tushargangurde2029
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/sample-todo
The recommended git tool is: NONE
using credential ubuntu
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/sample-todo/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/tushargangurde2029/django-todo.git # timeout=10
Fetching upstream changes from https://github.com/tushargangurde2029/django-todo.git
> git --version # timeout=10
> git --version # 'git version 2.34.1'
using GIT_SSH to set credentials lkjdsfl
Verifying host key using known hosts file
```

As you can see in the above image build successfully run by github after pushing code and also title has been change and new image is deployed successfully

TODO APP By :: Django

Title*

Value*

Date*