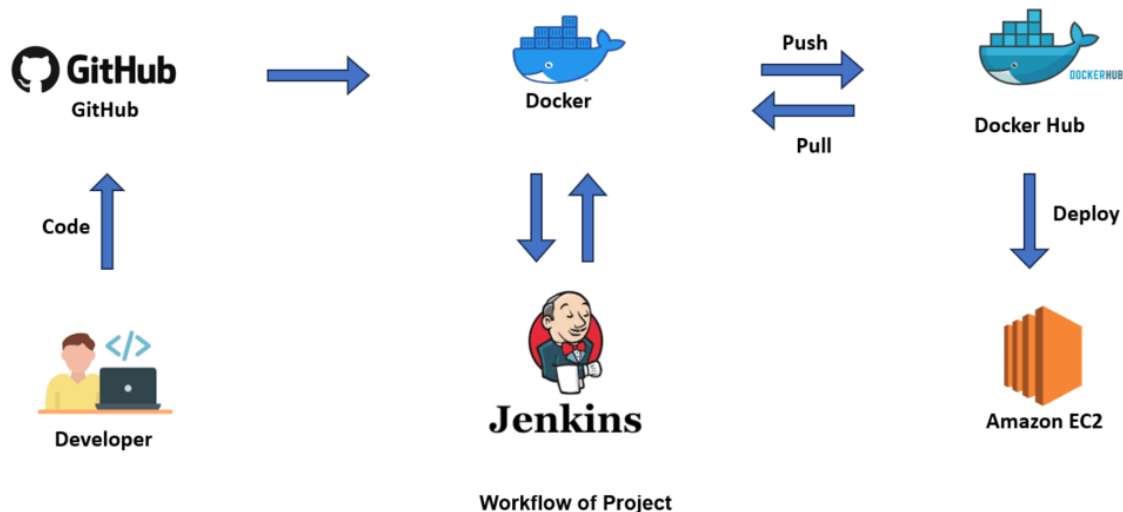


# Django Application to EC2 Instance using Jenkins

## Prerequisites:

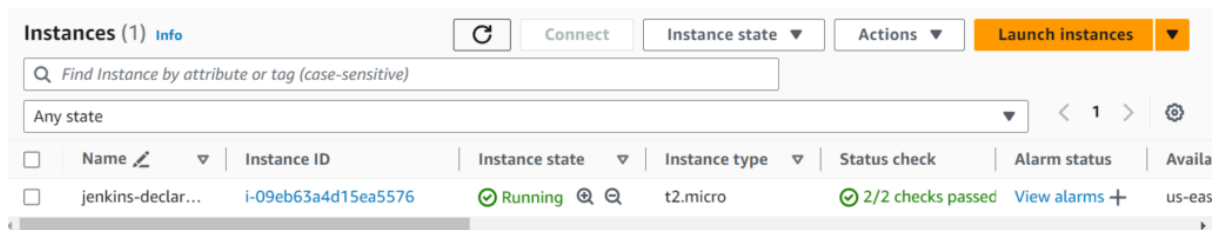
- AWS Account.
- Working GitHub Account.
- Docker Hub Account.

## Workflow:



## Steps:

### 1. Create an Ec2 instance with Linux AMI



### 2. Login to Ec2 Instance

Select the Server and click on Connect.

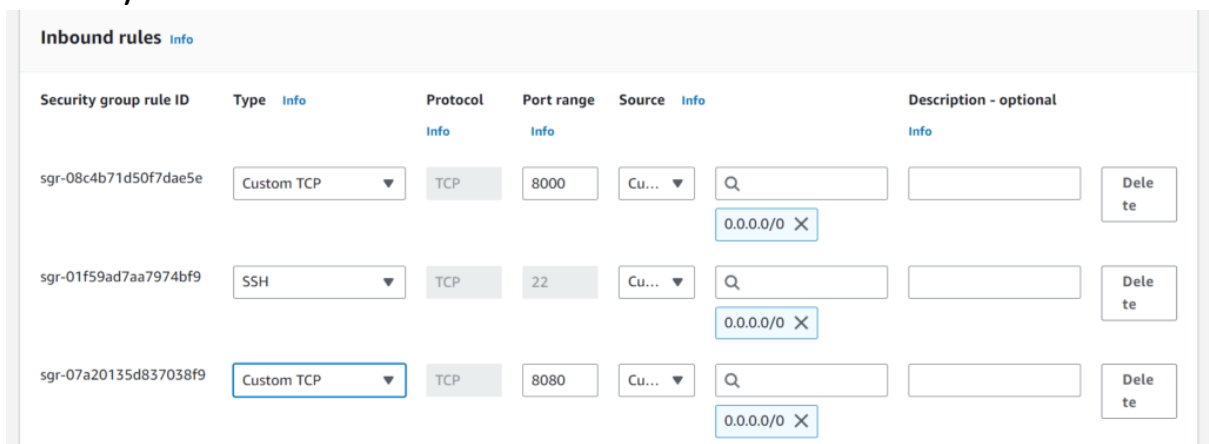
### 3. Install Git and Jenkins in EC2 Instance

By using the series of below commands you can install git and jenkins in ec2 Instance

```
#sudo yum install git -y
#sudo yum install java-17* -y
#sudo wget -O /etc/yum.repos.d/jenkins.repo
https://pkg.jenkins.io/redhat-stable/jenkins.repo
#sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
#sudo yum install jenkins -y
#sudo bash
#systemctl start jenkins
#systemctl enable jenkins
#systemctl status jenkins
#cat /var/lib/jenkins/secrets/initialAdminPassword
```

### 4. Login and Configure Plugins

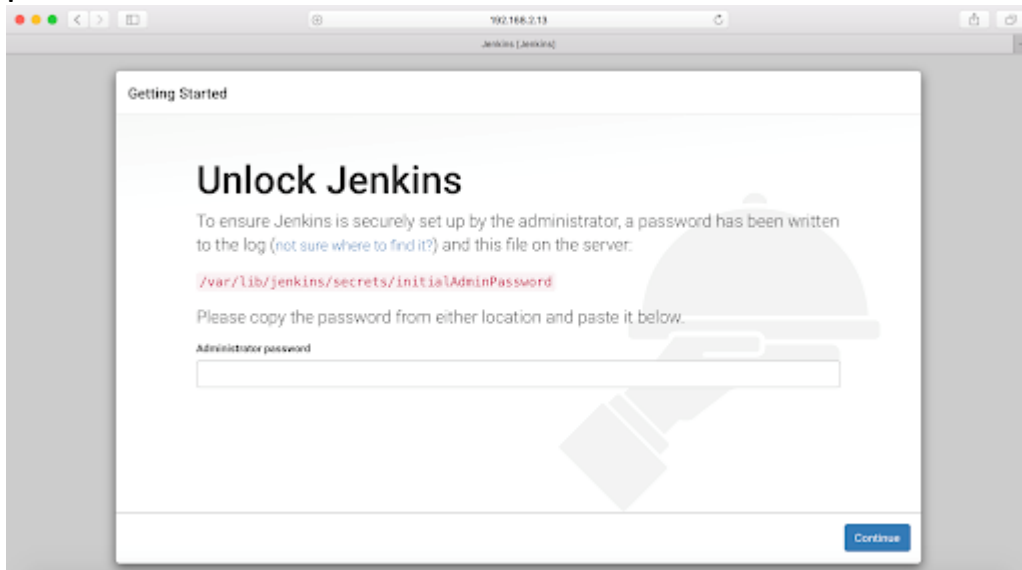
To login and configure we have to enable the port 8080 and 8000 in the server security group (SG)>>inbound rules>>edit>>add rule>>port 8080 and port 8000>>access 0.0.0.0/00>>Submit.



Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
sgr-08c4b71d50f7dae5e	Custom TCP	TCP	8000	Cu... 0.0.0.0/0		Delete
sgr-01f59ad7aa7974bf9	SSH	TCP	22	Cu... 0.0.0.0/0		Delete
sgr-07a20135d837038f9	Custom TCP	TCP	8080	Cu... 0.0.0.0/0		Delete

Open new tab in the browser and copy the Public IP address of the server and paste it in the new tab followed by “:8080” and

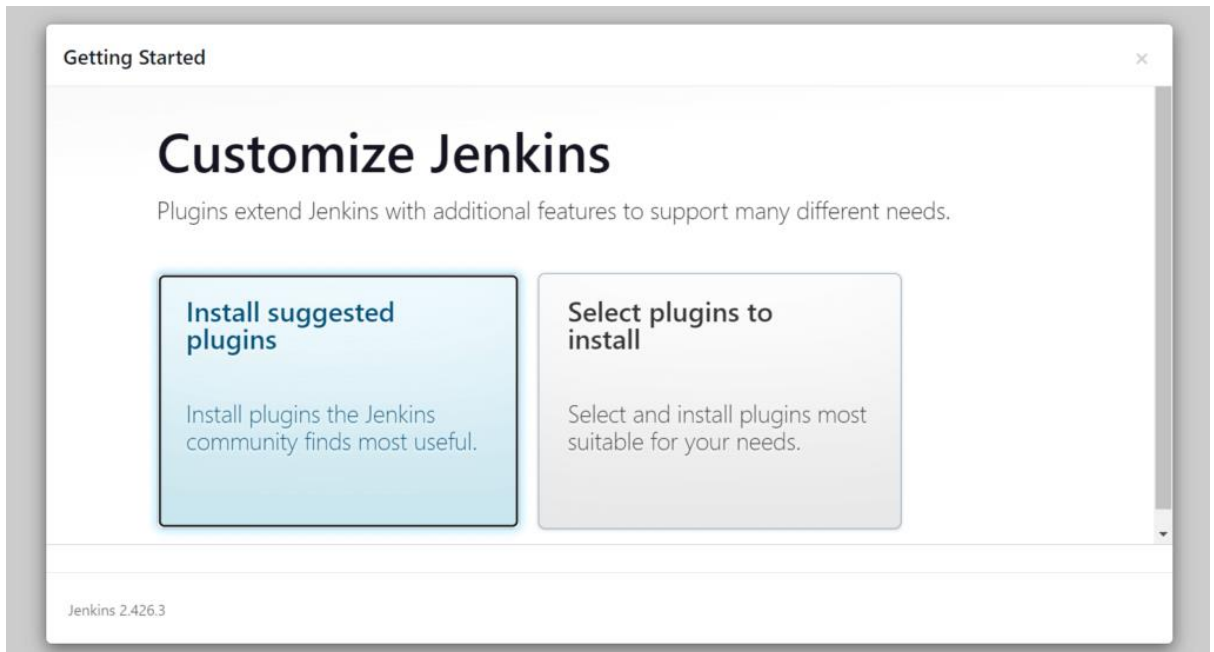
press enter



You'll see the above page.

In that paste the code which we get from this command (`#cat /var/lib/jenkins/secrets/initialAdminPassword`)

Give the necessary details and click on "save and continue".  
Leave it default and click "save and continue".



## 5. Install Docker and Give Permissions

Use the below command

```
# yum install docker -y
```

```
# systemctl start docker
# systemctl enable docker
# systemctl status Jenkins
# usermod -aG docker ec2-user
# chmod 666 /var/run/docker.sock
# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"
# unzip awscliv2.zip
```

## 6. Clone the Repository

```
#git clone <URL>
#ls
#cd Django-notes-app
#reboot
#cat Dockerfile
```

```
ubuntu@ip-172-31-41-214:~/django-notes-app$ ls
Dockerfile  README.md  db.sqlite3  manage.py  notesapp  requirements.txt
Jenkinsfile  api        docker-compose.yml  mynotes  procfile  staticfiles
ubuntu@ip-172-31-41-214:~/django-notes-app$ cat Dockerfile
FROM python:3.9

WORKDIR /app/backend

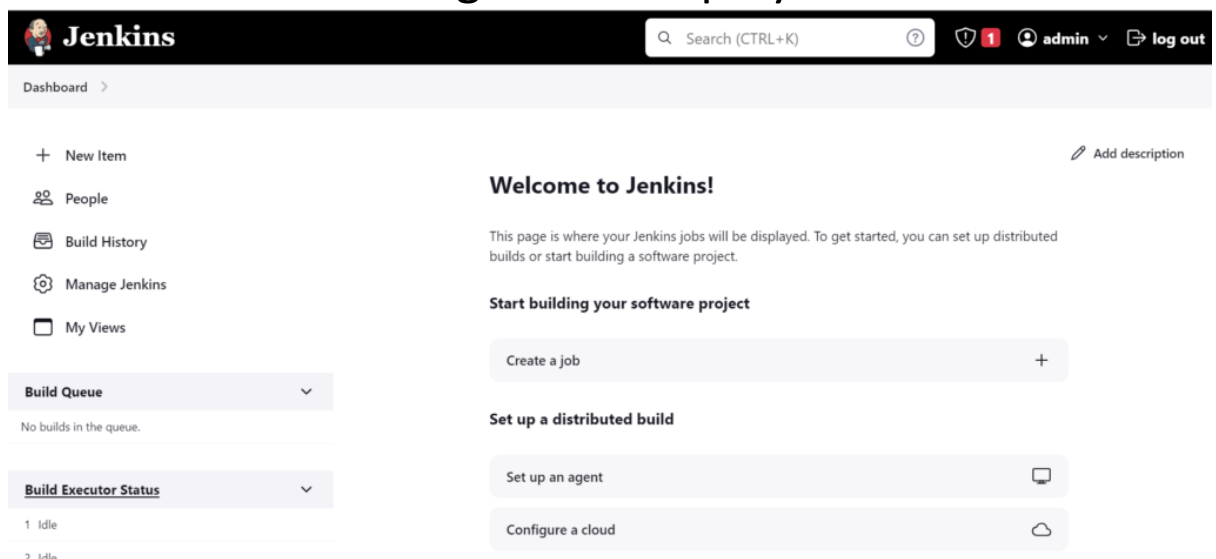
COPY requirements.txt /app/backend
RUN pip install -r requirements.txt

COPY . /app/backend

EXPOSE 8000

CMD python /app/backend/manage.py runserver 0.0.0.0:8000
ubuntu@ip-172-31-41-214:~/django-notes-app$ |
```

## 7. Create Job and Configure for Deployment



The screenshot shows the Jenkins web interface. At the top is a navigation bar with the Jenkins logo, a search bar, and user information (admin). The main content area is divided into a left sidebar and a main panel. The sidebar contains links for 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'My Views'. Below these are two expandable sections: 'Build Queue' (showing 'No builds in the queue') and 'Build Executor Status' (showing two idle executors). The main panel displays a 'Welcome to Jenkins!' message, explaining that this is where jobs are displayed and providing instructions on how to get started. It includes a 'Start building your software project' section with a 'Create a job' button, and a 'Set up a distributed build' section with buttons for 'Set up an agent' and 'Configure a cloud'.


## Go to New Item


Dashboard >


**Enter an item name**


notes-app-cicd

» 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  
A container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a


OK


## Give the name and select the “Pipeline” click Ok


Dashboard > notes-app-cicd > Configuration

**Configure**

 General

 Advanced Project Options

 Pipeline

**General** Enabled 

Description

This is CI/CD Pipeline of Notes App.

Plain text [Preview](#)

☐ Discard old builds ?

☐ Do not allow concurrent builds

☐ Do not allow the pipeline to resume if the controller restarts


☒ GitHub project


Project url ?


## Give the description of the project and given the git hub project “url” and attach webhooks in GitHub

Dashboard > notes-app-cicd > Configuration

**Configure**

 General

 Advanced Project Options

 Pipeline

**Build Triggers**

☐ Build after other projects are built ?

☐ Build periodically ?


☒ GitHub hook trigger for GITScm polling ?

☐ Poll SCM ?

☐ Quiet period ?

☐ Trigger builds remotely (e.g., from scripts) ?

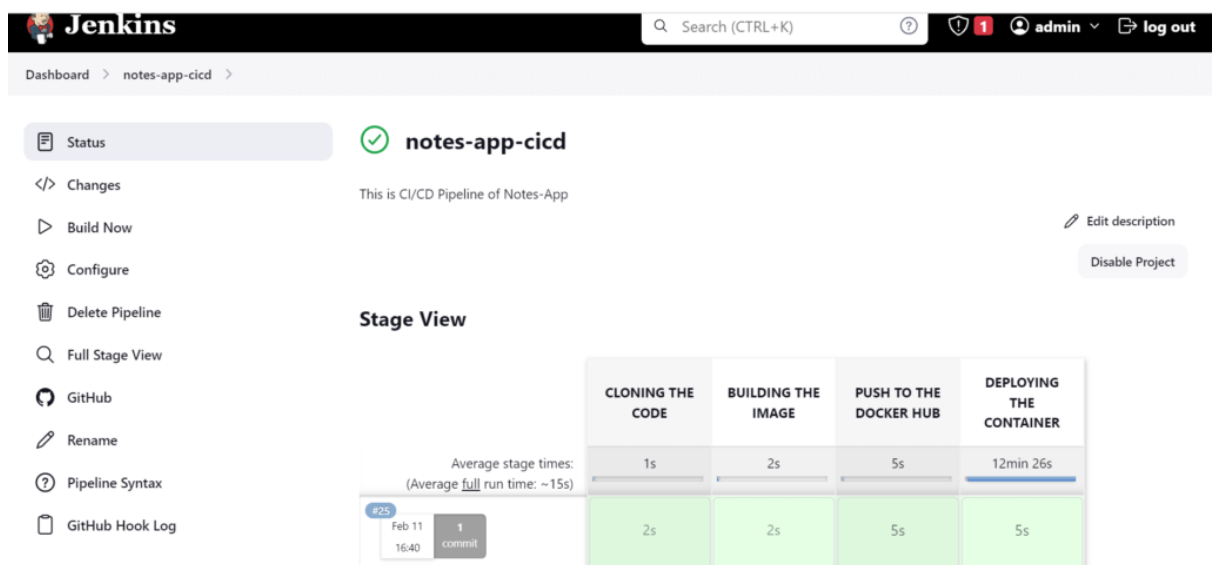
**Advanced Project Options**

Advanced 

Add credentials in Dashboards>>Manage Jenkins>>Credentials>>System>>Global Credentials>>Add Credentials in this add username, password, CredentialsID

Give the Pipeline code and modify the necessary modifications in “Jenkins file” in git hub

Then click on “save” and “apply”.  
Click on “Build Now”.



## 8. To Check the Application

Open a new tab in the Browser and paste the “public IP” address followed by “:8000”

