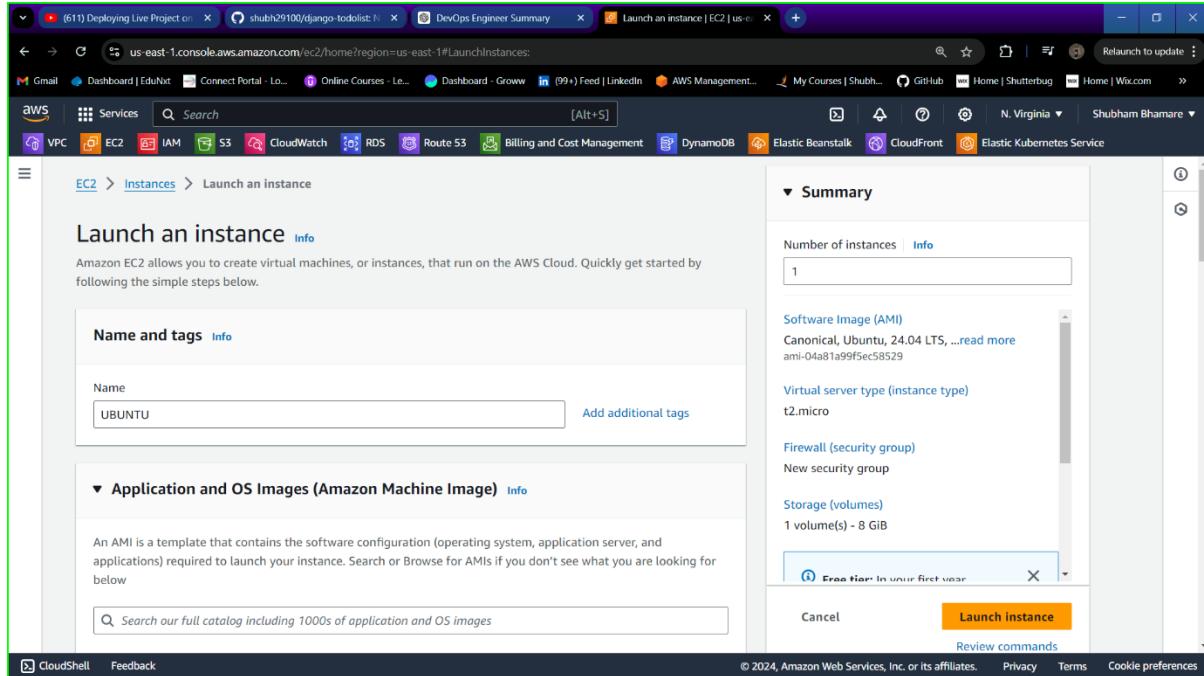


Create a CI/CD pipeline for web application using docker, Jenkins and AWS.

Step 1: Login into your AWS account. Go to EC2 dashboard and launch an instance (Allow SSH rule).



Step 2: Take remote access of your instance and run command “sudo apt-get update” to update the system.

A screenshot of a terminal window titled 'ubuntu@ip-172-31-90-90:~\$ sudo apt-get update'. The window shows the progress of the package update process, listing numerous packages being downloaded from various Ubuntu repositories. The terminal interface includes tabs for 'Sessions' and 'Tools', and a sidebar with session management options like 'New', 'Logout', and 'Logout'.

Step 3: Run “sudo apt install git -y” to install version control system (Github).

```
3.86.198.48 (ubuntu)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
[ 2 3.86.198.48 (ubuntu) ]
ubuntu@ip-172-31-98-95:~$ sudo apt install git -y
[sudo] password for ubuntu:
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.1).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 39 not upgraded.
ubuntu@ip-172-31-98-95:~$
```

UNREGISTERED VERSION - Please support Mobaxterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Step 4: Create a new directory using command “mkdir todo” , then go into that directory using command “cd todo” and then initialize git into that directory using command “git init”.

After initializing git clone the github repository in which the project is stored

“git clone <https://github.com/shubh29100/todolist.git> ”. After this the project is stored in your local repository.

```
3.86.198.48 (ubuntu)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
[ 2 3.86.198.48 (ubuntu) ]
ubuntu@ip-172-31-98-95:~$ ls
ubuntu@ip-172-31-98-95:~$ mkdir todo
ubuntu@ip-172-31-98-95:~$ cd todo/
ubuntu@ip-172-31-98-95:~/todo$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/ubuntu/todo/.git/
ubuntu@ip-172-31-98-95:~/todo$ git clone https://github.com/shubh29100/django-todolist.git
Cloning into 'django-todolist'...
remote: Enumerating objects: 91, done.
remote: Counting objects: 100% (91/91), done.
remote: Compressing objects: 100% (80/80), done.
remote: Writing objects: 100% (91/91), 110.89 KiB | 7.39 MiB/s, done.
Resolving deltas: 100% (22/22), done.
ubuntu@ip-172-31-98-95:~/todo$ ls
django-todolist
ubuntu@ip-172-31-98-95:~/todo$ cd django-todolist/
ubuntu@ip-172-31-98-95:~/todo/django-todolist$ ls
Dockerfile LICENSE README.md db.sqlite3 manage.py nohup.out staticfiles todoApp todos
ubuntu@ip-172-31-98-95:~/todo/django-todolist$
```

UNREGISTERED VERSION - Please support Mobaxterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Step 5: Run command “ sudo apt install python3-pip -y” to install python 3 in your EC2 instance.

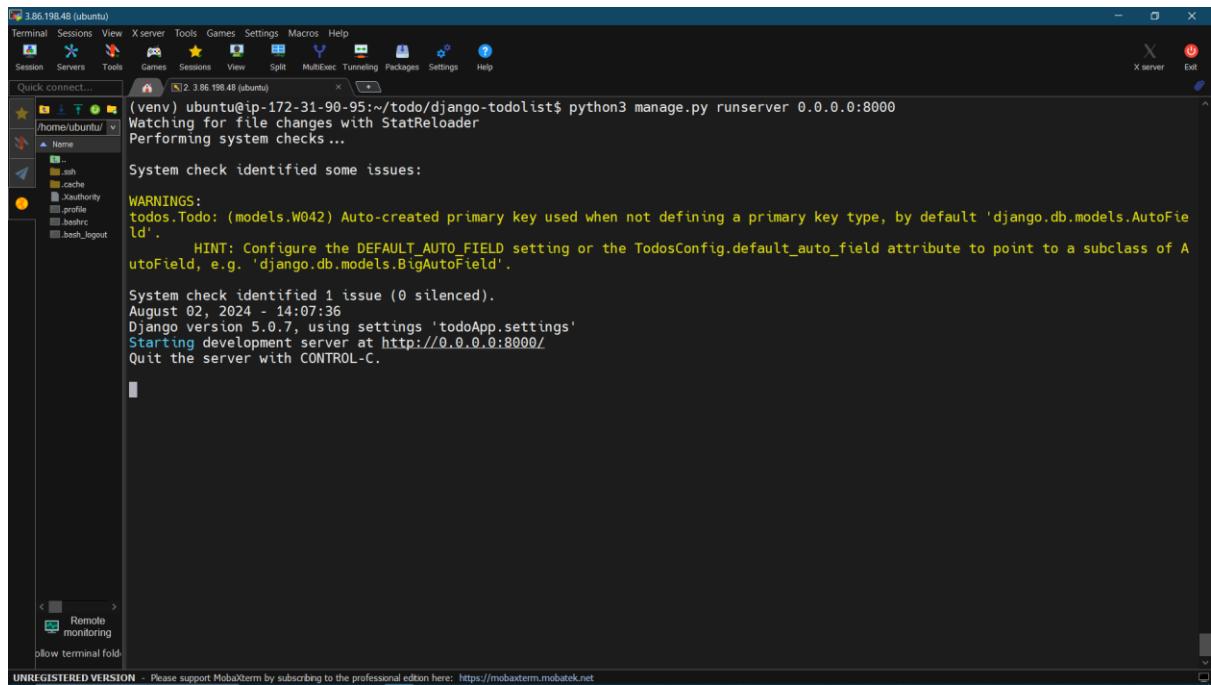
```
ubuntu@ip-172-31-96-95:~/todo/django-todolist$ sudo apt install python3-pip -y
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86_64-linux-gnu build-essential bztp2 cpp cpp-13 cpp-13-x86_64-linux-gnu cpp-x86_64-linux-gnu dpkg-dev fakeroot fontconfig-config fonts-dejavu-core fonts-dejavu-mono g++ g++-13 g++-13-x86_64-linux-gnu g++-x86_64-linux-gnu gcc-13 gcc-13-base gcc-13-x86_64-linux-gnu gcc-x86_64-linux-gnu
  javascript-common libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libatomic1 libbinutils libc-dev-bin libc-devtools
  libgcc-13-dev libgcc1-0 libcrypt-dev libcrypt-nobfd0 libctf0 libde265-0 libdeflate0 libdpkg-perl libexpat1-dev libfakeroot libfile-fcntllock-perl libfontconfig
  libgcc-13-dev libgdb1 libgomp1 libgprofng libheif-plugin-aomenc libheif-plugin-lbde265 libheif1 libwasan0 libis123 libitm libjbig0
  libjpeg-turbo8 libjs-jquery libjs-sphinxdoc libjsunderscore liblberc4 liblsan0 libmpc3 libpython3-dev libpython3.12-dev libpython3.12-minimal
  libstdc++-13-dev libtiff6 libtsan2 libwebp libxml2 libxpm4 linux-libc-dev lto-disabled-list make manpages-dev python3-dev python3-wheel python3.12-dev
  rpcsvc-proto zlib1g-dev
Suggested packages:
  binutils-doc gprofng-gui bztp2-doc cpp-doc gcc-13-locales cpp-13-doc debian-keyring g++-multilib g++-13-multilib gcc-13-doc gcc-multilib autoconf automake libtool
  flex bison gdc-doc gcc-13-multilib gdb-x86_64-linux-gnu apache2 | lighttpd | libpplgbc-doc bzr libgd-tools libheif-plugin-x86_65 libheif-plugin-ffmpegdec
  libheif-plugin-jpegdec libheif-plugin-jpgenc libheif-plugin-j2kdec libheif-plugin-j2kenc libheif-plugin-rav1e libheif-plugin-svtenc libstdc++-13-doc make-doc
  python3.12-venv python3.12-doc binfmt-support
The following NEW packages will be installed:
  binutils binutils-common binutils-x86_64-linux-gnu build-essential bztp2 cpp cpp-13 cpp-13-x86_64-linux-gnu cpp-x86_64-linux-gnu dpkg-dev fakeroot fontconfig-config
  fonts-dejavu-core fonts-dejavu-mono g++ g++-13 g++-13-x86_64-linux-gnu g++-x86_64-linux-gnu gcc-13 gcc-13-base gcc-13-x86_64-linux-gnu gcc-x86_64-linux-gnu
  javascript-common libalgorithm-diff-perl libalgorithm-merge-perl libatomic1 libbinutils libc-dev-bin libc-devtools
  libgcc-13-dev libgcc1-0 libcrypt-dev libcrypt-nobfd0 libctf0 libde265-0 libdeflate0 libdpkg-perl libexpat1-dev libfakeroot libfile-fcntllock-perl libfontconfig
  libgcc-13-dev libgdb1 libgomp1 libgprofng libheif-plugin-aomenc libheif-plugin-lbde265 libheif1 libwasan0 libis123 libitm libjbig0
  libjpeg-turbo8 libjs-jquery libjs-sphinxdoc libjsunderscore liblberc4 liblsan0 libmpc3 libpython3-dev libpython3.12-dev libquadmath0 libssframe1 libsharpuyuv
  libstdc++-13-dev libtiff6 libtsan2 libwebp libxml2 libxpm4 linux-libc-dev lto-disabled-list make manpages-dev python3-dev python3-wheel python3.12-dev
  rpcsvc-proto zlib1g-dev
The following packages will be upgraded:
  libpython3.12-stdlib libpython3.12tlib libpython3.12t64 linux-tools-common python3.12 python3.12-minimal
6 upgraded, 0 newly installed, 0 to remove and 33 not upgraded.
Need to get 97.2 MB of archives.
After this operation, 328 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libpython3.12t64 amd64 3.12.3-1ubuntu0.1 [2339 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 python3.12 amd64 3.12.3-1ubuntu0.1 [651 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libpython3.12-stdlib amd64 3.12.3-1ubuntu0.1 [2069 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 python3.12-minimal amd64 3.12.3-1ubuntu0.1 [2334 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libpython3.12-minimal amd64 3.12.3-1ubuntu0.1 [832 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 binutils-common amd64 2.42-4ubuntu2 [239 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libframe1 amd64 2.42-4ubuntu2 [14.8 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libframe1 amd64 2.42-4ubuntu2 [14.8 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libctf0 amd64 2.42-4ubuntu2 [9.1 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libtff0 amd64 2.42-4ubuntu2 [94.5 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libprofng0 amd64 2.42-4ubuntu2 [851 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 binutils-x86_64-linux-gnu amd64 2.42-4ubuntu2 [2469 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 binutils amd64 2.42-4ubuntu2 [18.0 kB]
```

Step 6: now we have create virtual environment to run project “python3 -m venv venv”. After this run command “source venv/bin/activate” to activate the virtual environment.

Now run command “pip install Django” to install Django into your virtual environment.

```
ubuntu@ip-172-31-90-95:~/todo/django-todolist$ python3 -m venv venv
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$ source venv/bin/activate
Collecting django
  Downloading Django-5.0.7-py3-none-any.whl.metadata (4.1 kB)
Collecting asigref<4, >=3.7.0 (from django)
  Downloading asigref-3.8.1-py3-none-any.whl.metadata (9.3 kB)
Collecting sqlparse>0.3.1 (from django)
  Downloading sqlparse-0.5.1-py3-none-any.whl.metadata (3.9 kB)
  Downloading Django-5.0.7-py3-none-any.whl (8.2 MB)
    8.2/8.2 MB 60.1 MB/s eta 0:00:00
  Downloading asigref-3.8.1-py3-none-any.whl (23 kB)
  Downloading sqlparse-0.5.1-py3-none-any.whl (44 kB)
    44.2/44.2 kB 4.3 MB/s eta 0:00:00
Installing collected packages: sqlparse, asigref, django
Successfully installed asigref-3.8.1 django-5.0.7 sqlparse-0.5.1
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$ django-admin --version
5.0.7
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$
```

Step 7: Run command “python3 manage.py runserver 0.0.0.0:8000” to start your project on port no 8000 .



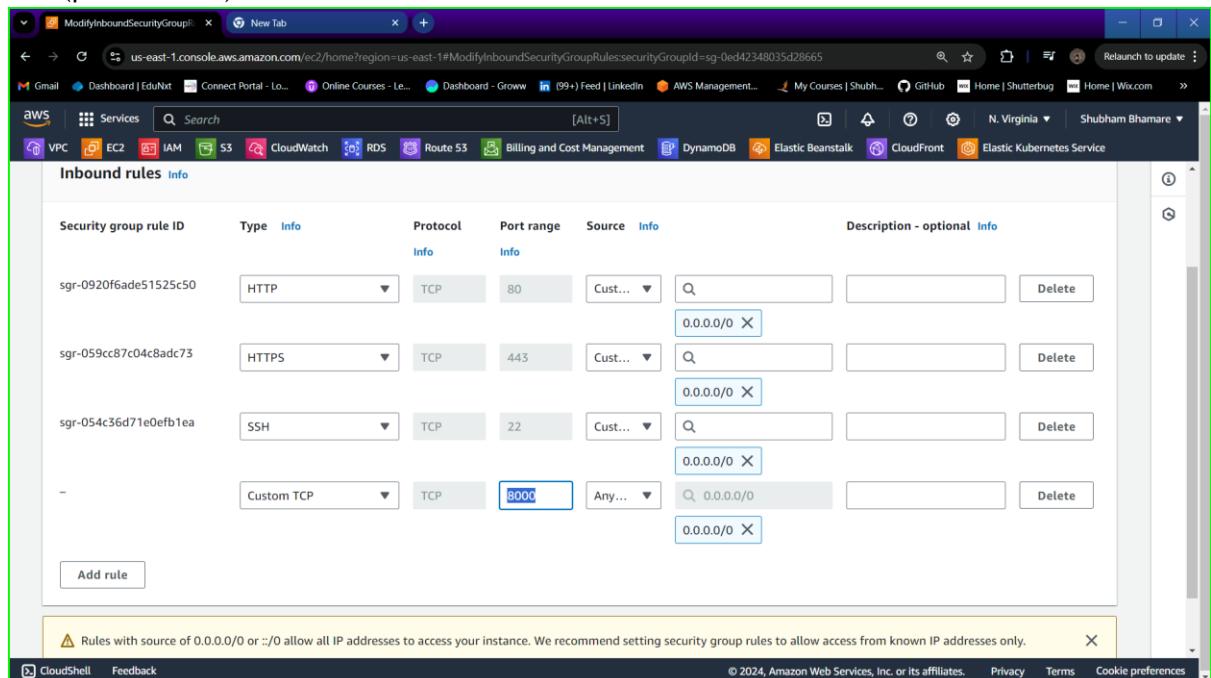
```
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$ python3 manage.py runserver 0.0.0.0:8000
Watching for file changes with StatReloader
Performing system checks...

System check identified some issues:

WARNINGS:
todos.Todo: (models.W042) Auto-created primary key used when not defining a primary key type, by default 'django.db.models.AutoField'.
    HINT: Configure the DEFAULT_AUTO_FIELD setting or the TodosConfig.default_auto_field attribute to point to a subclass of AutoField, e.g. 'django.db.models.BigAutoField'.

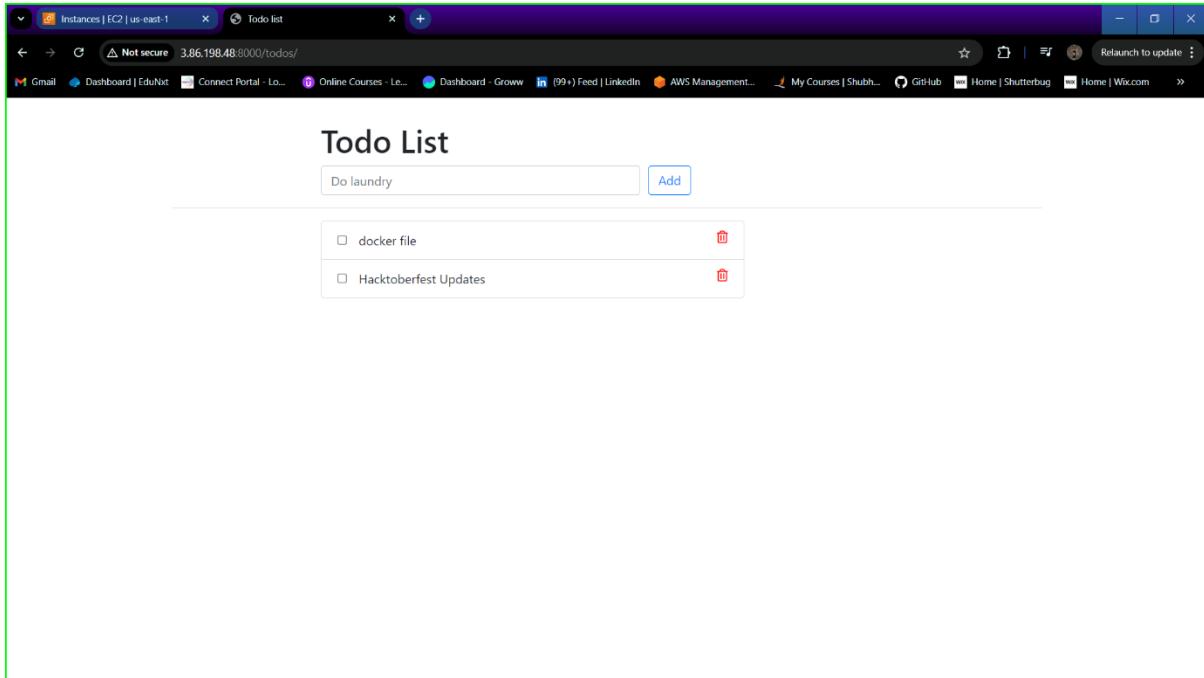
System check identified 1 issue (0 silenced).
August 02, 2024 - 14:07:36
Django version 5.0.7, using settings 'todoApp.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
```

Step 8: Go to your security group which is attached to your EC2 Instance and add a rule for custom TCP (port no 8000) and save it.



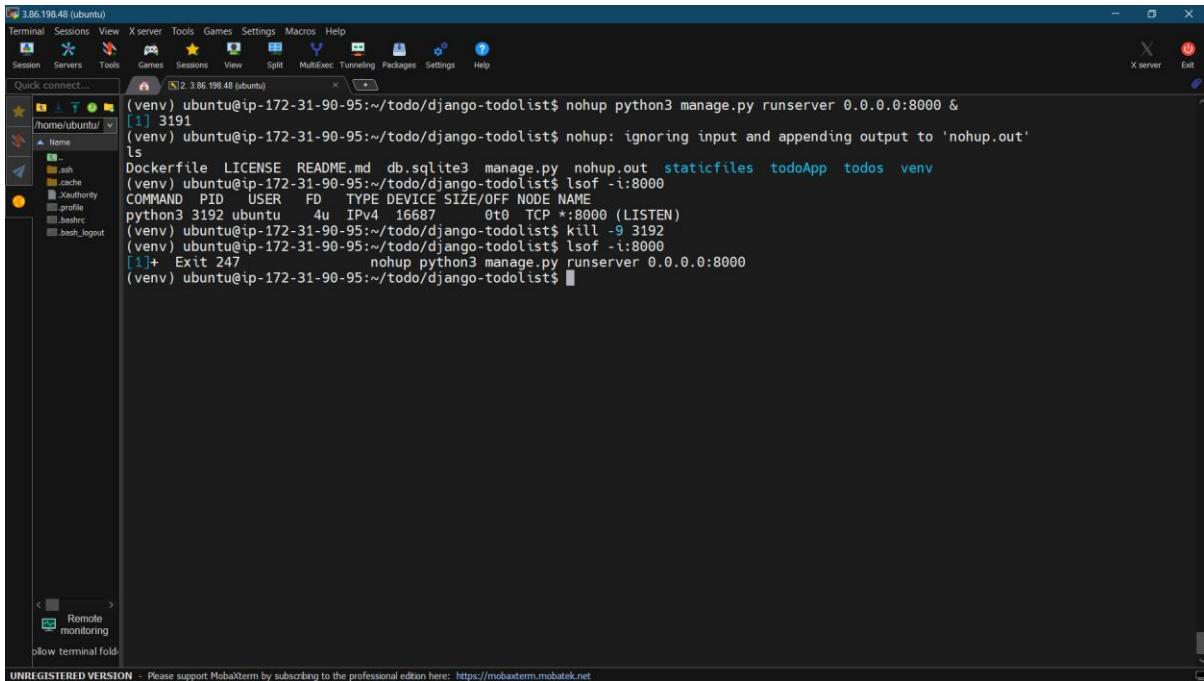
Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0920f6ade51525c50	HTTP	TCP	80	Cust...	0.0.0.0/0
sgr-059cc87c04c8adc73	HTTPS	TCP	443	Cust...	0.0.0.0/0
sgr-054c36d71e0efb1ea	SSH	TCP	22	Cust...	0.0.0.0/0
-	Custom TCP	TCP	8000	Any...	0.0.0.0/0

Step 9: Now run “public Ip:8080” and you can that the project is running successfully.



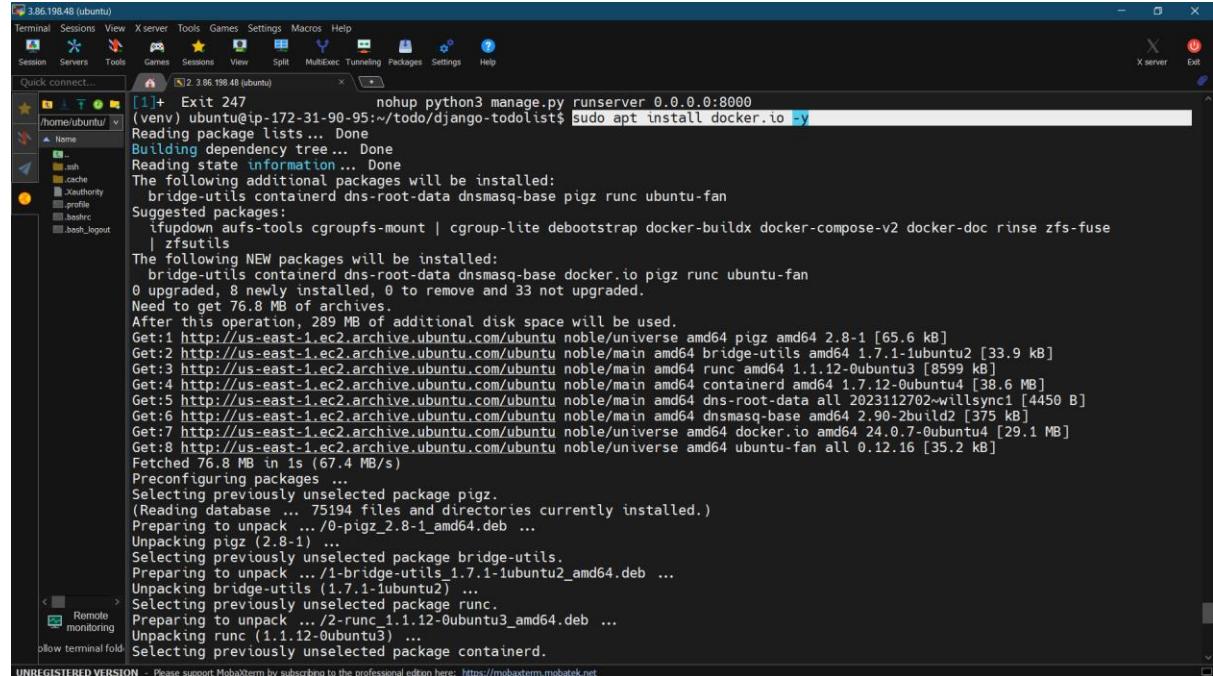
Step 10: Run command “nohup python3 manage.py runserver 0.0.0.0:8000 &” so that the project will run in background and you can use the terminal.

Run command “lsof -i:8000” to see which service is running on you 8000 port. Then run command “kill -9 PID” to kill that service.



Step 11: Now you have to install docker into your server to create a docker image which we later require to automate the process.

Run command “ sudo apt install docker.io -y” to install docker.



```
[1]+  Exit 247          nohup python3 manage.py runserver 0.0.0.0:8000
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$ sudo apt install docker.io -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-buildx docker-compose-v2 docker-doc rinse zfs-fuse
| zfsutils
The following NEW packages will be installed:
bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 33 not upgraded.
Need to get 76.8 MB of archives.
After this operation, 289 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 bridge-utils amd64 1.7.1-1ubuntu2 [33.9 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 runc amd64 1.1.12-0ubuntu3 [8599 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 containerd amd64 1.7.12-0ubuntu4 [38.6 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 dns-root-data all 2023112702-willsync1 [4450 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 dnsmasq-base amd64 2.90-2build2 [375 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 docker.io amd64 24.0.7-0ubuntu4 [29.1 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 ubuntu-fan all 0.12.16 [35.2 kB]
Fetched 76.8 MB in 1s (67.4 MB/s)
Preconfiguring packages...
Selecting previously unselected package pigz.
(Reading database ... 75194 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package bridge-utils.
Preparing to unpack .../1-bridge-utils_1.7.1-1ubuntu2_amd64.deb ...
Unpacking bridge-utils (1.7.1-1ubuntu2) ...
Selecting previously unselected package runc.
Preparing to unpack .../2-runc_1.1.12-0ubuntu3_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu3) ...
Selecting previously unselected package containerd.
```

Step 12: Create a new Docker file ,Run command “vim Dockerfile” .

Press “I” to get into insert mode and then write

“ FROM python:3

RUN pip install django==3.2

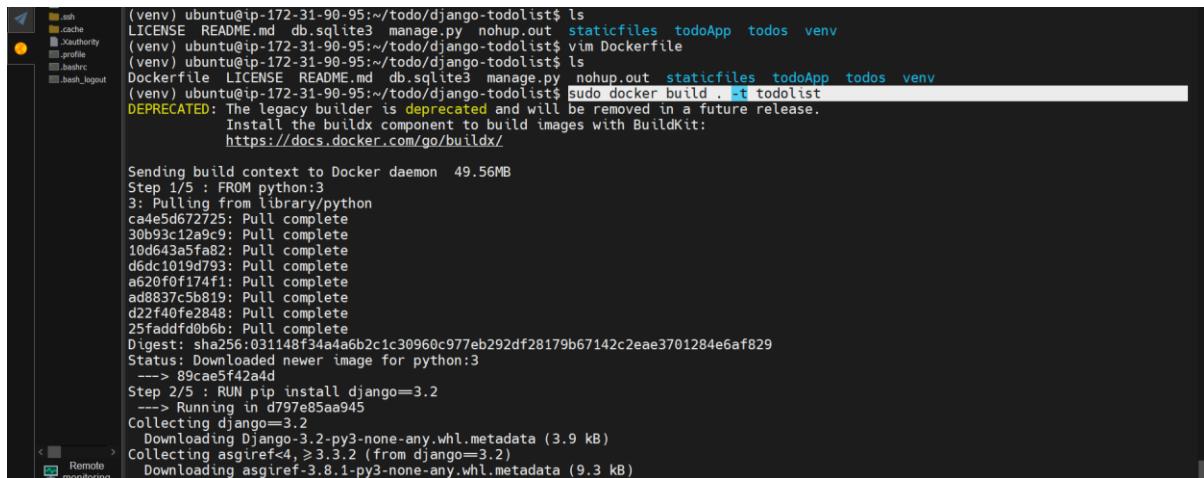
COPY . .

RUN python manage.py migrate

CMD [“python”,“manage.py”,“runserver”,“0.0.0.0:8000”] “

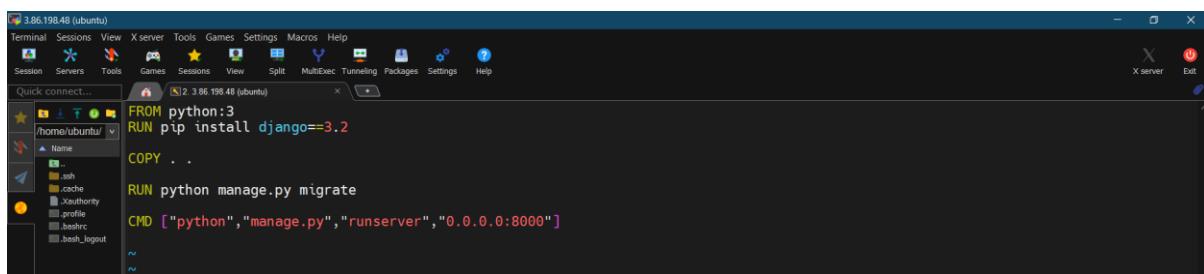
Then press “esc” to come out of insert mode and the press “:wq!” to save the file.

After saving Dockerfile run command "sudo docker build . -t todolist" to build a image from dockerfile.



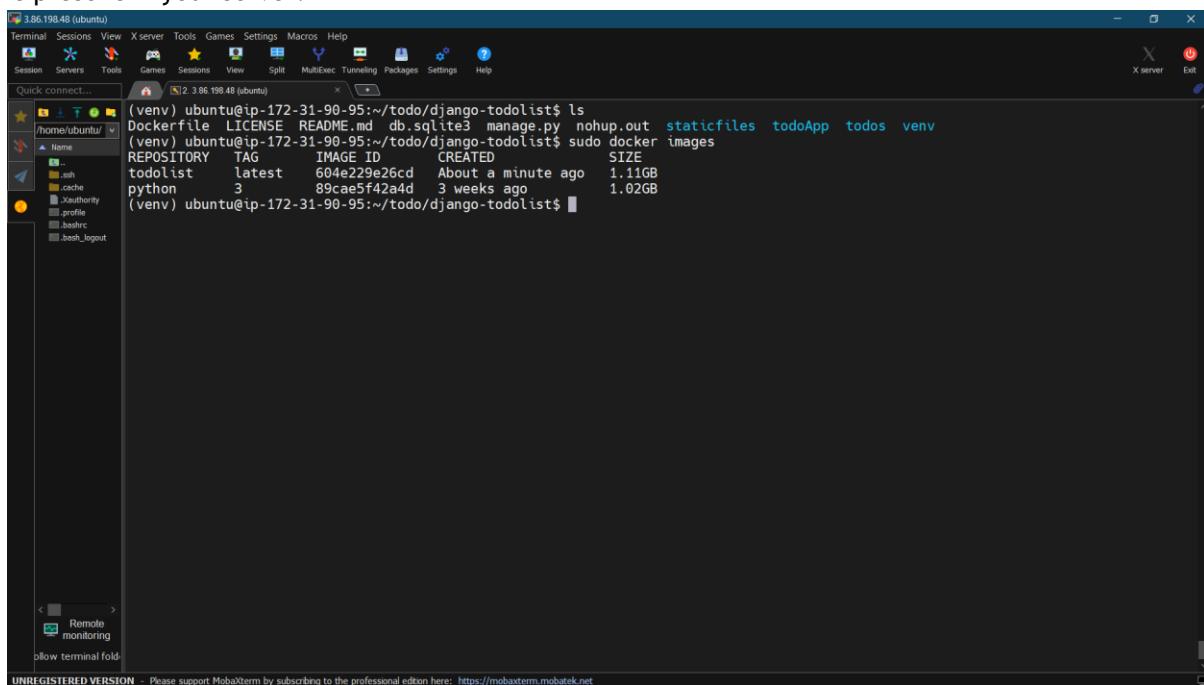
```
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$ ls
LICENSE README.md db.sqlite3 manage.py nohup.out staticfiles todoApp todos venv
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$ vim Dockerfile
Dockerfile LICENSE README.md db.sqlite3 manage.py nohup.out staticfiles todoApp todos venv
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$ sudo docker build . -t todolist
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/
Sending build context to Docker daemon 49.56MB
Step 1/5 : FROM python:3
3: Pulling from library/python
ca4e5d672725: Pull complete
30b93c12a9c9: Pull complete
10d643a5fa82: Pull complete
d6dc1019d793: Pull complete
a620f0f174f1: Pull complete
ad8837c5b819: Pull complete
d22f40fe2848: Pull complete
25fadff0b6b6: Pull complete
Digest: sha256:031148f34a4a6b2c1c30960c977eb292df28179b67142c2eae3701284e6af829
Status: Downloaded newer image for python:3
--> 89cae5f42a4d
Step 2/5 : RUN pip install django==3.2
--> Running in d797e85aa945
Collecting django==3.2
  Downloading Django-3.2-py3-none-any.whl.metadata (3.9 kB)
Collecting asgiref<4,>3.3.2 (from django==3.2)
  Downloading asgiref-3.8.1-py3-none-any.whl.metadata (9.3 kB)


```



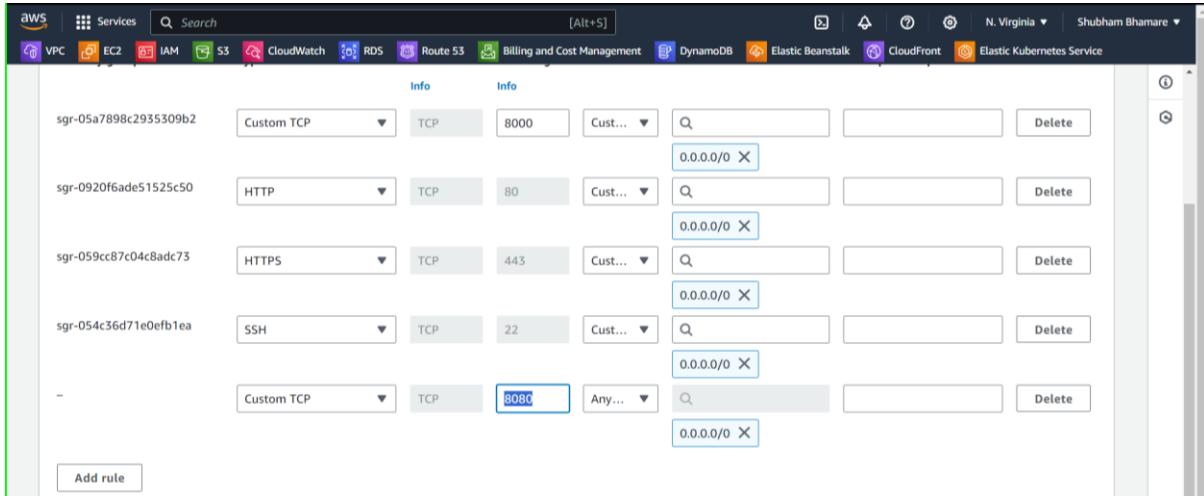
```
FROM python:3
RUN pip install django==3.2
COPY .
RUN python manage.py migrate
CMD ["python",""manage.py"",""runserver"",""0.0.0.0:8000""]
```

Step 13: The image is built successfully. Run command "sudo docker images" to check if the image is present in your server.



```
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$ ls
Dockerfile LICENSE README.md db.sqlite3 manage.py nohup.out staticfiles todoApp todos venv
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$ sudo docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
todolist latest 604e229e26cd About a minute ago 1.11GB
python    3     89cae5f42a4d 3 weeks ago   1.02GB
(venv) ubuntu@ip-172-31-90-95:~/todo/django-todolist$
```

Step 14: Now go to your security group again and add rule for 8080 port and save it.



Step 15: Now you have install java and Jenkins on your server.

Run command “sudo apt install openjdk-11-jdk -y” to install java.

“java -version” to check java version.

Now run “curl -fsSL https://pkg.jenkins.io/debian/jenkins.io.key | sudo tee \ /usr/share/keyrings/jenkins-keyring.asc > /dev/null” to import Jenkins GPG key.

Run “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” - adds Jenkins repository to your system.

Run “ sudo apt update” - Update your package list again to include the new Jenkins repository.

Run “sudo apt install jenkins -y” to install Jenkins.

Run “sudo systemctl start Jenkins” to start Jenkins service.

Run “sudo systemctl enable Jenkins” - enable Jenkins to start on boot.

Run “`sudo ufw allow 8080`” - to adjust firewall.

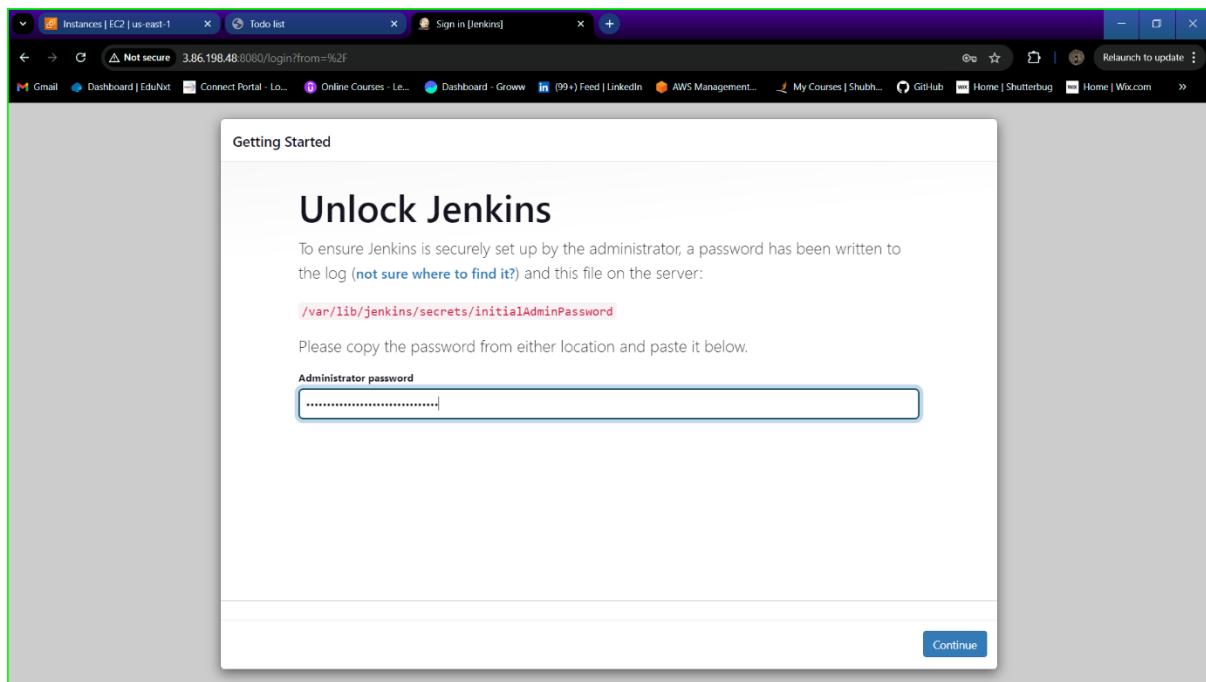
```
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
Session 7.386.198.48 (ubuntu)
/home/ubuntu/ ~
Name
jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: active (running) since Fri 2024-08-02 10:17:09 UTC; 1min 8s ago
       Main PID: 9307 (java)
         Tasks: 37 (limit: 1130)
        Memory: 284.3M (peak: 315.8M)
          CPU: 1min 1.289s
        CGroup: /system.slice/jenkins.service
               └─9307 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Aug 02 10:16:20 ip-172-31-90-95 jenkins[9307]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Aug 02 10:16:20 ip-172-31-90-95 jenkins[9307]: ****
Aug 02 10:17:09 ip-172-31-90-95 jenkins[9307]: 2024-08-02 10:17:09.950+0000 [id=30] INFO jenkins.InitReactorRunner$!#onAttained: Completed 8
Aug 02 10:17:09 ip-172-31-90-95 jenkins[9307]: 2024-08-02 10:17:09.990+0000 [id=22] INFO hudson.lifecycle.Lifecycle$onReady: Jenkins is full
Aug 02 10:17:09 ip-172-31-90-95 jenkins[9307]: 2024-08-02 10:17:10.133+0000 [id=45] INFO h.m.DownloadService$Downloadable$load: Obtained the
Aug 02 10:17:09 ip-172-31-90-95 jenkins[9307]: 2024-08-02 10:17:10.136+0000 [id=45] INFO hudson.util.Retrier#start: Performed the action ch
[lines 1-20/20 (END)] ... skipping ...
jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: active (running) since Fri 2024-08-02 10:17:09 UTC; 1min 8s ago
       Main PID: 9307 (java)
         Tasks: 37 (limit: 1130)
        Memory: 284.3M (peak: 315.8M)
          CPU: 1min 1.289s
        CGroup: /system.slice/jenkins.service
               └─9307 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

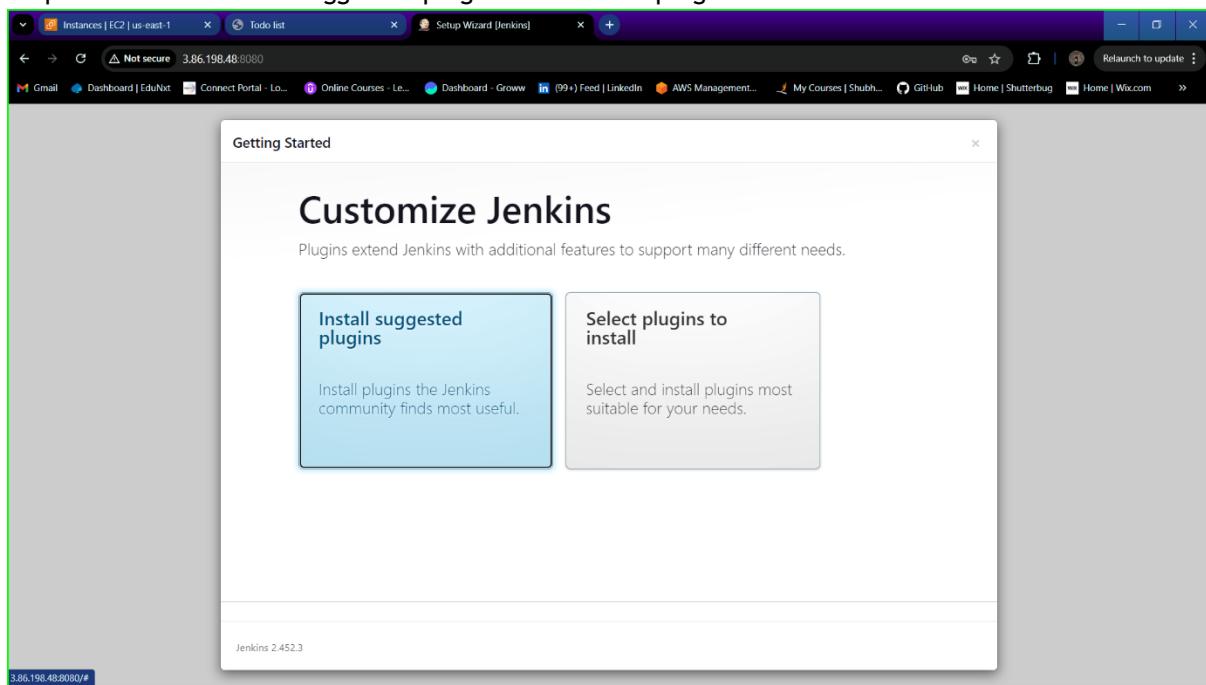
Aug 02 10:16:20 ip-172-31-90-95 jenkins[9307]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Aug 02 10:16:20 ip-172-31-90-95 jenkins[9307]: ****
Aug 02 10:16:20 ip-172-31-90-95 jenkins[9307]: ****
Aug 02 10:16:20 ip-172-31-90-95 jenkins[9307]: ****
Aug 02 10:17:09 ip-172-31-90-95 jenkins[9307]: 2024-08-02 10:17:09.950+0000 [id=30] INFO jenkins.InitReactorRunner$!#onAttained: Completed 8
Aug 02 10:17:09 ip-172-31-90-95 jenkins[9307]: 2024-08-02 10:17:09.990+0000 [id=22] INFO hudson.lifecycle.Lifecycle$onReady: Jenkins is full
Aug 02 10:17:09 ip-172-31-90-95 jenkins[9307]: 2024-08-02 10:17:10.133+0000 [id=45] INFO h.m.DownloadService$Downloadable$load: Obtained the
Aug 02 10:17:09 ip-172-31-90-95 jenkins[9307]: 2024-08-02 10:17:10.136+0000 [id=45] INFO hudson.util.Retrier#start: Performed the action ch
```

Step 16: Run command “`sudo cat /var/lib/jenkins/secrets/initialAdminPassword`” to view Jenkins password.

Step 17: To access Jenkins go to browser and run “<PublicIp>:8080” and pate the Jenkins initial password.



Step 18: Click on Install suggested plugins and let the plugins install.



Step 19: Now we have to create User to access Jenkins fill in all details and click on save and continue.

The screenshot shows a browser window with the URL <http://3.86.198.48:8080>. The title bar says "Setup Wizard [Jenkins]". The main content area is titled "Create First Admin User". It contains five input fields: "Username" (shubham), "Password" (empty), "Confirm password" (empty), "Full name" (shubham bhamare), and "E-mail address" (shubhamara29@gmail.com). At the bottom right are two buttons: "Skip and continue as admin" and "Save and Continue".

Step 20: Click on save and continue.

The screenshot shows a browser window with the URL <http://3.86.198.48:8080>. The title bar says "Setup Wizard [Jenkins]". The main content area is titled "Instance Configuration". It has one input field "Jenkins URL" with the value "http://3.86.198.48:8080/". Below it is a detailed description of what the Jenkins URL is used for. At the bottom right are two buttons: "Not now" and "Save and Finish".

Step 21: Now create one node named “todoapp”.

The screenshot shows the Jenkins agent configuration page for a node named 'todoapp'. The page includes sections for 'Status', 'Delete Agent', 'Configure', 'Build History', 'Load Statistics', and 'Log'. It also contains command-line instructions for running from the agent's command line on Unix and Windows, and an option to run from a secret file. A 'Mark this node temporarily offline' button is visible in the top right corner.

Step 22: Now you have to add your github credentials to Jenkins. For that you have to go to “manage Jenkins” tab.

The screenshot shows the Jenkins dashboard with the 'Manage Jenkins' tab selected. The page features a 'Welcome to Jenkins!' message, a 'Start building your software project' section, and a 'Build Queue' and 'Build Executor Status' section. The 'Build Executor Status' section lists two nodes: 'Built-In Node' (offline) and 'todoapp' (offline). The URL '3.86.198.48:8080/manage' is visible at the bottom of the page.

Step 23: Then click on “System”.

The screenshot shows the Jenkins System Configuration page. At the top, there is a warning message: "Java 11 end of life in Jenkins" with options to "More Info" or "Ignore". Below this, the "System Configuration" section is visible, featuring several tabs: "System" (selected), "Tools", "Plugins", "Nodes", "Clouds", "Appearance", "Security", "Credentials", and "Users". The "System" tab has a sub-section titled "Configure global settings and paths." The "Security" tab includes "Security" (Secure Jenkins; define who is allowed to access/use the system) and "Users" (Create/delete/modify users that can log in to this Jenkins). The "Credentials" tab shows a dropdown menu with "none" selected and a "+ Add" button. The "Advanced" section at the bottom contains "Save" and "Apply" buttons.

Step 24: Scroll down until you find Github section. Then add a github sever,

In credentials click on “Add” and then click on Jenkins.

The screenshot shows the GitHub configuration page under the "System" section. It displays the "GitHub Servers" configuration. A new "GitHub Server" is being added, with the "Name" field set to "Github", the "API URL" field containing "https://api.github.com", and the "Credentials" dropdown showing "none". A "Test connection" button is present. The "Manage hooks" checkbox is unchecked. At the bottom, there are "Save" and "Apply" buttons.

Step 25: Select Kind- Secret Text

Scope- Global

Secret: Enter your github secret token here.

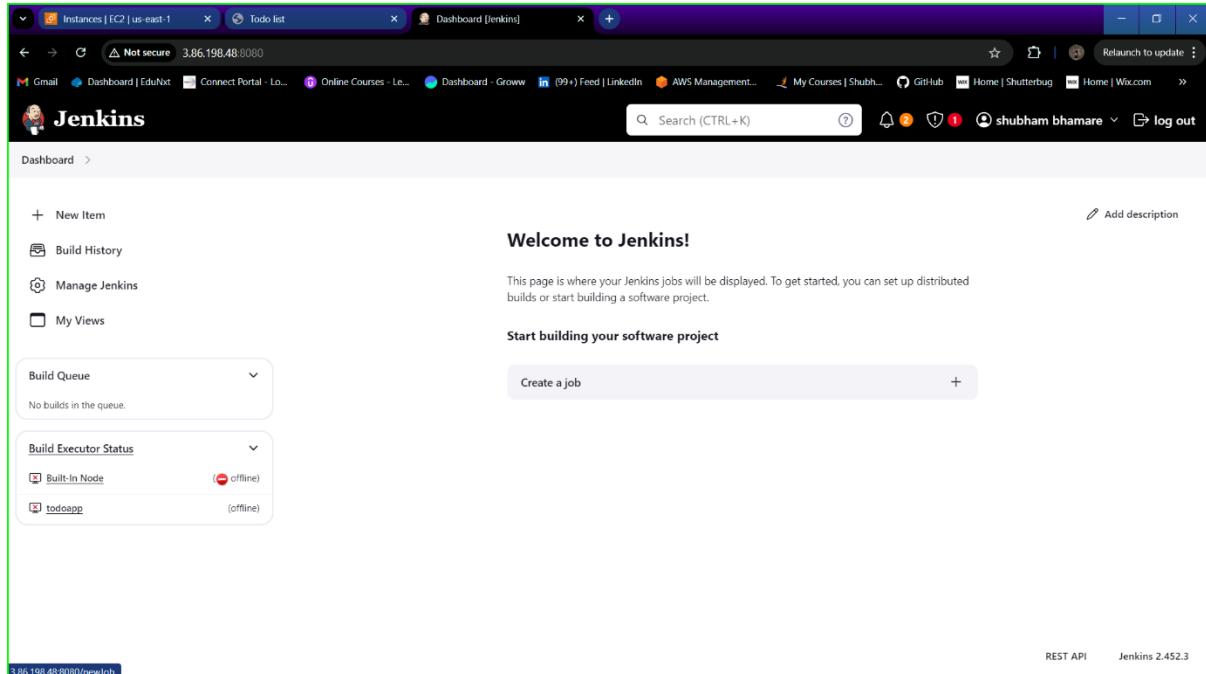
Id- Enter the name to your credentials and then save it.

The screenshot shows the Jenkins 'Add Credentials' page. The 'Kind' is set to 'Secret text'. The 'Scope' is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Secret' field contains a masked password. The 'ID' field is set to 'Git-cicd'. The 'Description' field is empty. At the bottom, there are 'Save' and 'Apply' buttons.

Step 26: Then again in credentials select your credentials and save.

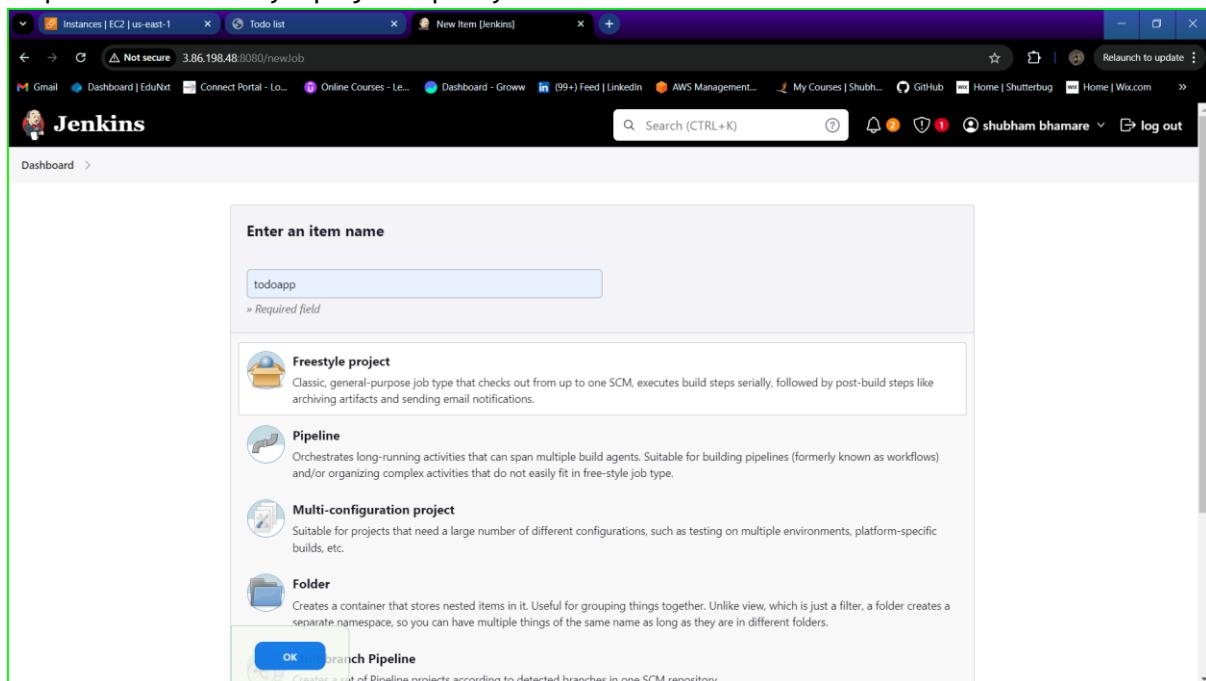
The screenshot shows the 'GitHub Server' configuration page. Under 'Credentials', the dropdown menu shows 'todoapp' selected. There is a 'Test connection' button at the bottom right. At the very bottom, there are 'Save' and 'Apply' buttons.

Step 27: Then come to dashboard and click on “New Item”.



The screenshot shows the Jenkins dashboard at the URL 3.86.198.48:8080. The main header says "Welcome to Jenkins!". On the left sidebar, there is a "New Item" button under the "Dashboard" section. The central area has sections for "Build Queue" and "Build Executor Status". The "Build Queue" section shows "No builds in the queue." The "Build Executor Status" section shows two nodes: "Built-In Node" (offline) and "todoapp" (offline). At the bottom right, it says "REST API Jenkins 2.452.3".

Step 28: Select Freestyle project. Specify the name and click on “ok”.



The screenshot shows the "New Item" creation dialog at the URL 3.86.198.48:8080/newJob. The title bar says "New Item [jenkins]". The main form has a field "Enter an item name" containing "todoapp", which is marked as a "Required field". Below this, there are four project types listed: "Freestyle project" (selected), "Pipeline", "Multi-configuration project", and "Folder". Each type has a brief description. At the bottom of the dialog, there is an "OK" button and a link "Branch Pipeline".

Step 29: Scroll down and paste your github repository link in “Repository URL”.

In credentials- leave it blank.

The screenshot shows the Jenkins configuration interface for a job named 'todoapp'. Under the 'Source Code Management' section, the 'Git' option is selected. In the 'Repositories' section, there is one entry with the 'Repository URL' set to `https://github.com/shubh29100/django-todolist.git`. The 'Credentials' dropdown is set to '- none -'. There are 'Save' and 'Apply' buttons at the bottom.

Step 30: Check your branch name.

The screenshot shows the Jenkins configuration interface for a job named 'todoapp'. Under the 'Branches to build' section, the 'Branch Specifier (blank for 'any')' field contains `*/main`. There are 'Save' and 'Apply' buttons at the bottom.

Step 31: In Build Step: Select “Execute Shell”.

The screenshot shows the Jenkins configuration interface for a job named 'todoapp'. The 'Build Steps' section is open, displaying a dropdown menu with various options: 'Execute Windows batch command', 'Execute shell' (which is highlighted with a light blue background), 'Invoke Ant', 'Invoke Gradle script', 'Invoke top-level Maven targets', 'Run with timeout', and 'Set build status to "pending" on GitHub commit'. The 'Execute shell' option is currently selected.

Step 32: Enter the commands

“sudo docker build -t todo-app” and “sudo docker run -p 8000:8000 -d todo-app”.

Then click on “save”.

The screenshot shows the Jenkins configuration interface for the 'todoapp' job. The 'Build Steps' section now contains a single step labeled 'Execute shell'. The 'Command' field contains the following two lines of Docker commands:
sudo docker build . -t todo-app
sudo docker run -p 8000:8000 -d todo-app

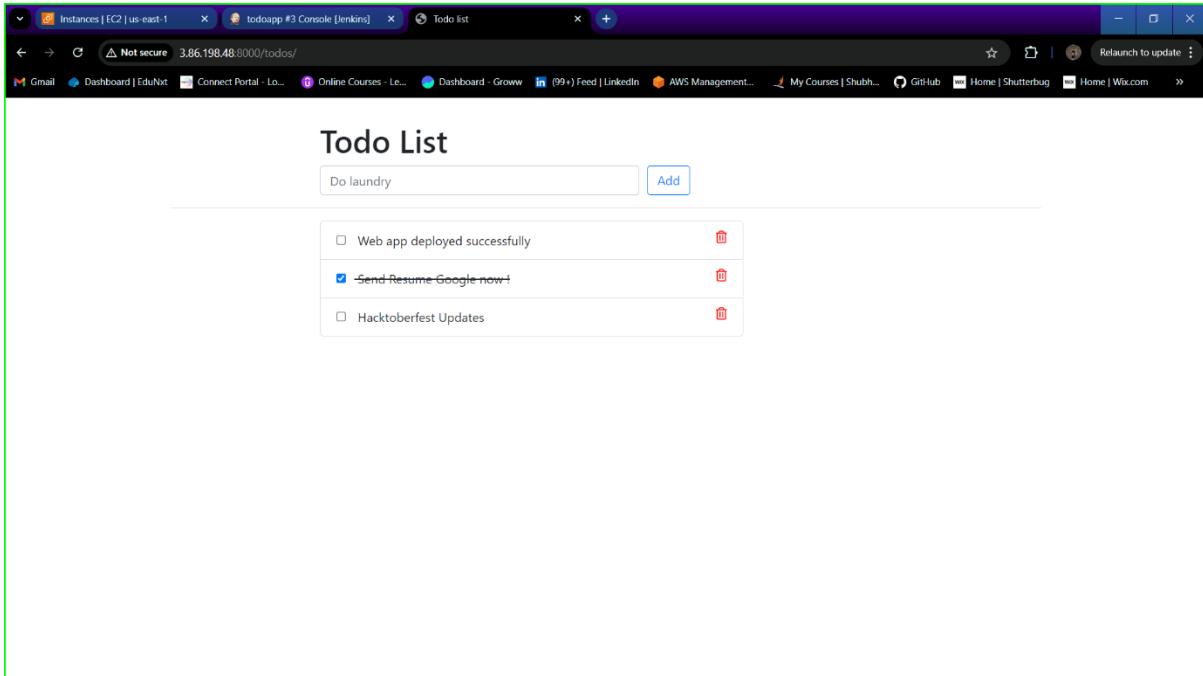
Step 33: Job is created successfully. Now click on build now to build your job.

The screenshot shows the Jenkins interface for the 'todoapp' project. On the left, a sidebar lists options like Status, Changes, Workspace, Build Now (which is highlighted), Configure, Delete Project, and Rename. The main area is titled 'todoapp' and contains a 'Permalinks' section with links for 'Atom feed for all' and 'Atom feed for failures'. At the bottom, there's a 'Build History' section with a message 'No builds' and a 'trend' dropdown. The status bar at the bottom right indicates '3.86.198.48:8080/job/todoapp/build?delay=0sec' and 'Jenkins 2.452.3'.

Step 34: Job is completed successfully.

The screenshot shows the Jenkins console output for build #3. The sidebar on the left includes 'Status', 'Changes', 'Console Output' (which is selected), 'View as plain text', 'Edit Build Information', 'Delete build #3', 'Timings', 'Git Build Data', and 'Previous Build'. The main content area is titled 'Console Output' with a green checkmark icon. It displays the command-line logs of the build process, starting with 'Started by user shubham bhamare' and detailing the git fetch, checkout, and docker build steps. The status bar at the bottom right indicates '3.86.198.48:8080/job/todoapp/3/console' and 'Jenkins 2.452.3'.

Step 35: Application is running successfully.



Step 36: You can also check that one docker container is created by Jenkins for your application.

