Glassfish:

- Install Glassfish server and change HTTP port to 8088.
- Create a demo Java (11) servlet application with maven.
- Generate war packages.
- Deploy the war using glassfish app server.

Glassfish server is installed in Ubuntu 20.04, also default jdk package is installed which is default java development package.

'***Sudo apt update***'

sudo apt install default-jdk

Then an environment variable is set as:

nano /etc/environment

And this part is added to the file :

JAVA_HOME="/usr/lib/jvm/java-11-openjdk-amd64"

Source /etc/environment to load the variable.

Then latest Glassfish.zip file is downloaded using command:

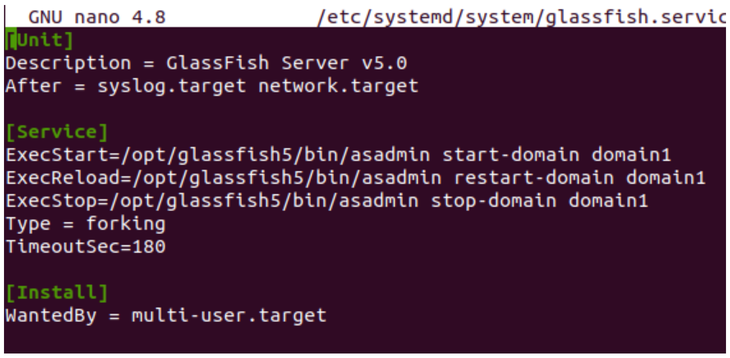
'wget http://download.oracle.com/glassfish/5.0.1/nightly/latest-glassfish.zip'

The file is then extracted

Sudo unzip latest-glassfish.zip

The glassfish.service file is created:
Sudo nano /etc/systemd/system/glassfish.service

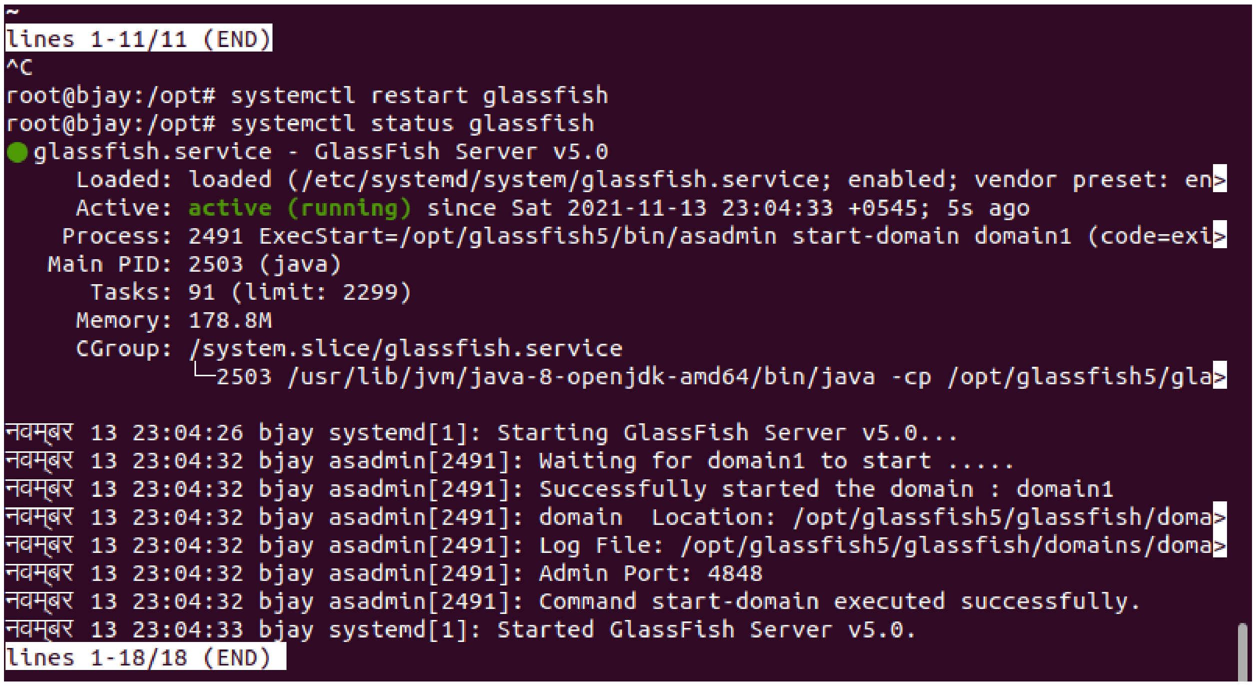
Inside that file, this part is appended:



```
  GNU nano 4.8             /etc/systemd/system/glassfish.servic
[Unit]
Description = GlassFish Server v5.0
After = syslog.target network.target

[Service]
ExecStart=/opt/glassfish5/bin/asadmin start-domain domain1
ExecReload=/opt/glassfish5/bin/asadmin restart-domain domain1
ExecStop=/opt/glassfish5/bin/asadmin stop-domain domain1
Type = forking
TimeoutSec=180

[Install]
WantedBy = multi-user.target
```

Then the daemon was reloaded and following are commands to start glassfish:
sudo systemctl daemon-reload
sudo systemctl enable glassfish
sudo systemctl start glassfish
sudo systemctl status glassfish

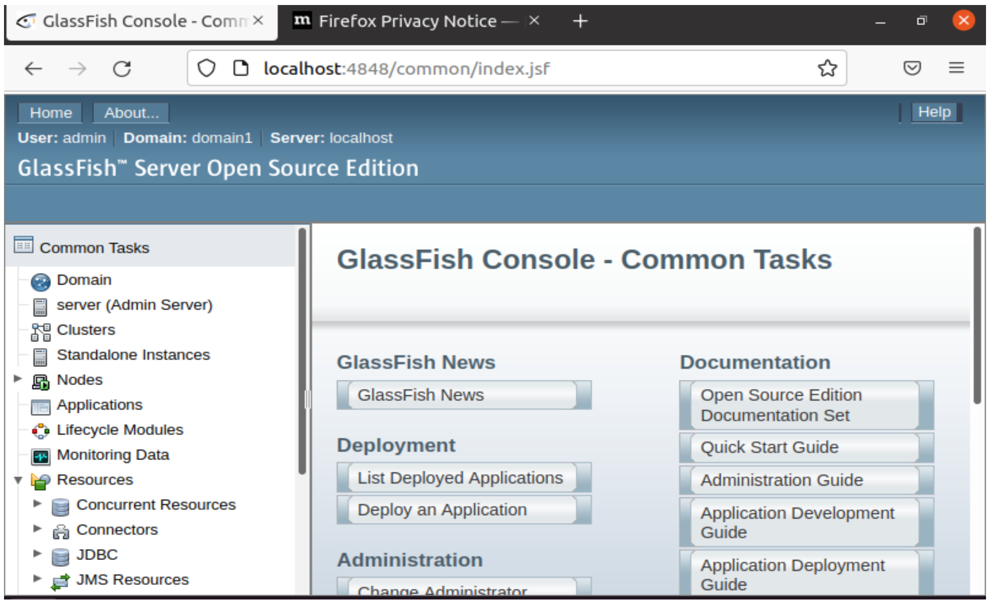
After this, we can see the status of glassfish:



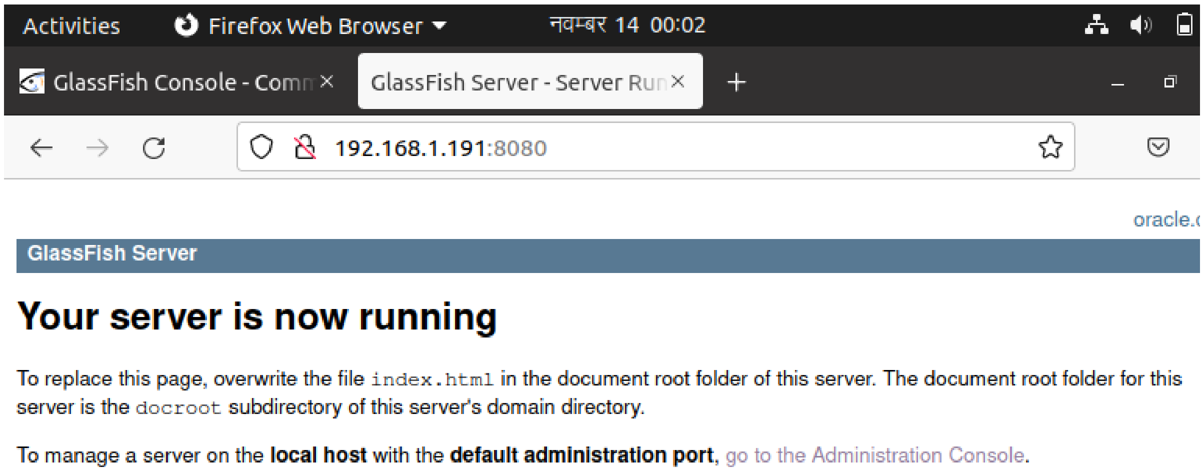
```
~
lines 1-11/11 (END)
^C
root@bjay:/opt# systemctl restart glassfish
root@bjay:/opt# systemctl status glassfish
● glassfish.service - GlassFish Server v5.0
     Loaded: loaded (/etc/systemd/system/glassfish.service; enabled; vendor preset: en>
     Active: active (running) since Sat 2021-11-13 23:04:33 +0545; 5s ago
    Process: 2491 ExecStart=/opt/glassfish5/bin/asadmin start-domain domain1 (code=exi>
   Main PID: 2503 (java)
      Tasks: 91 (limit: 2299)
     Memory: 178.8M
     CGroup: /system.slice/glassfish.service
             └─2503 /usr/lib/jvm/java-8-openjdk-amd64/bin/java -cp /opt/glassfish5/gla>

नवम्बर 13 23:04:26 bjay systemd[1]: Starting GlassFish Server v5.0...
नवम्बर 13 23:04:32 bjay asadmin[2491]: Waiting for domain1 to start .....
नवम्बर 13 23:04:32 bjay asadmin[2491]: Successfully started the domain : domain1
नवम्बर 13 23:04:32 bjay asadmin[2491]: domain  Location: /opt/glassfish5/glassfish/doma>
नवम्बर 13 23:04:32 bjay asadmin[2491]: Log File: /opt/glassfish5/glassfish/domains/doma>
नवम्बर 13 23:04:32 bjay asadmin[2491]: Admin Port: 4848
नवम्बर 13 23:04:32 bjay asadmin[2491]: Command start-domain executed successfully.
नवम्बर 13 23:04:33 bjay systemd[1]: Started GlassFish Server v5.0.
lines 1-18/18 (END)
```

After the glassfish server is active, we can access the admin panal with port 4848.

If we go to a browser and enter localhost:4848 then we can see this screen:



To go to the launch the app, we can enter the IP of the VM.
I.e. 192.168.1.191 in our case and default http port for glassfish service is 8080.
So when we go to 192.168.1.191:8080 we can launch the app,



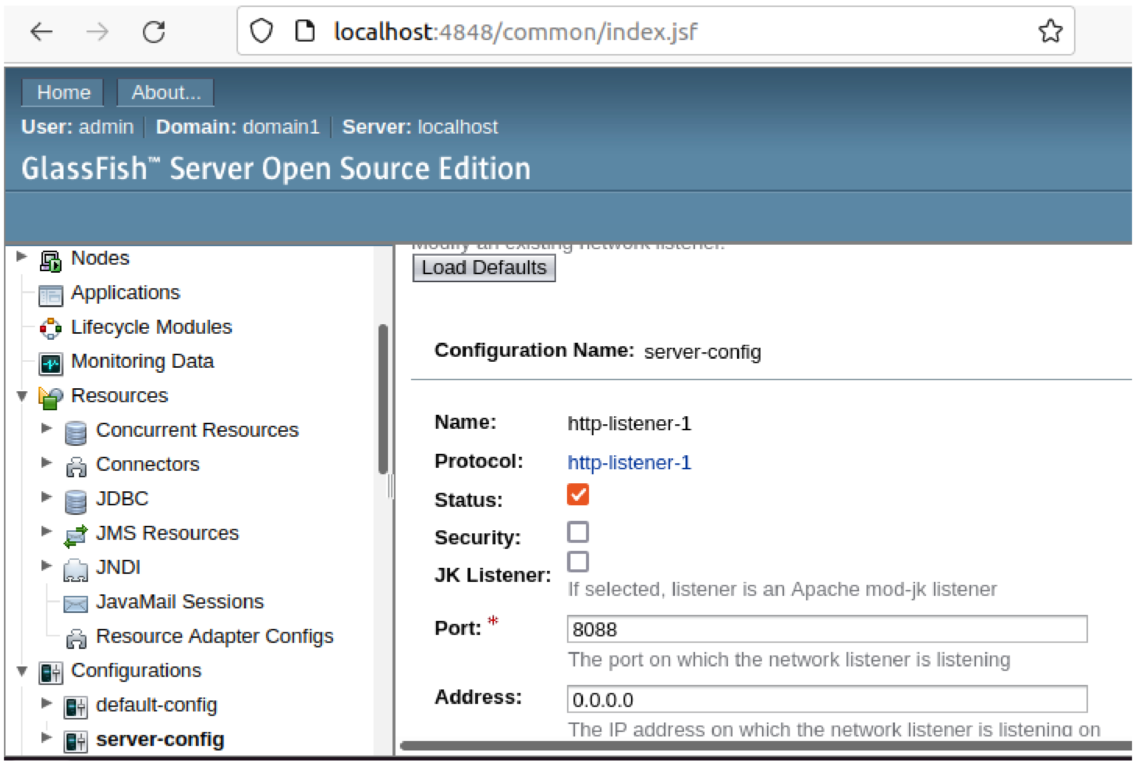
GlassFish Server

Your server is now running

To replace this page, overwrite the file index.html in the document root folder of this server. The document root folder for this server is the docroot subdirectory of this server's domain directory.

To manage a server on the **local host** with the **default administration port**, go to the Administration Console.

The default HTTP port of the glassfish server is: 8080
And the http port is changed from the admin server.
In the left side of admin server, we can see server config:



User: admin **Domain:** domain1 **Server:** localhost

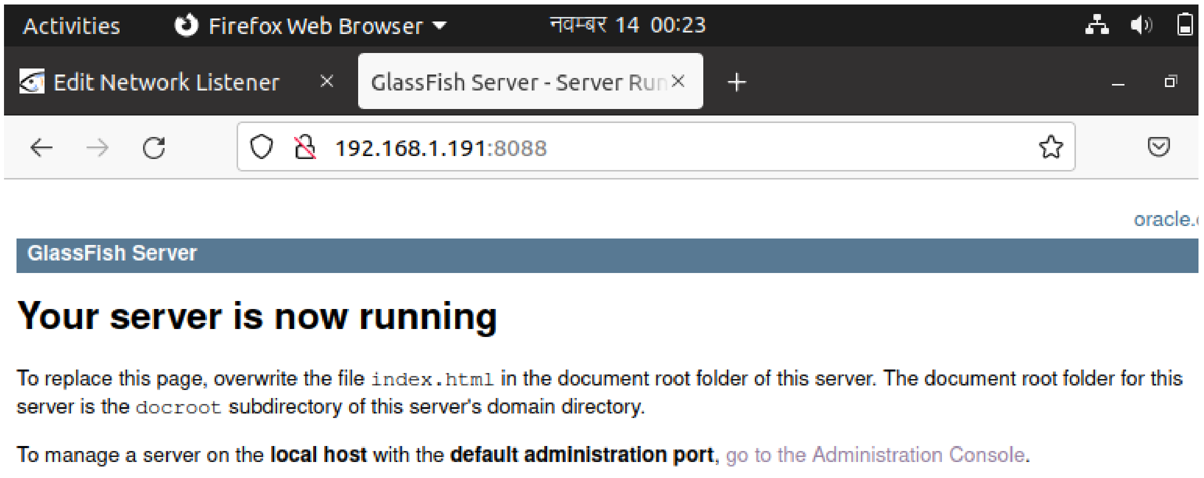
GlassFish™ Server Open Source Edition

Configuration Name: server-config

Name: http-listener-1
Protocol: http-listener-1
Status: ☑
Security: ☐
JK Listener: ☐ If selected, listener is an Apache mod-jk listener
Port: * 8088
The port on which the network listener is listening
Address: 0.0.0.0
The IP address on which the network listener is listening on

Inside server-config >NetworkListeners>http-listener-1
We changed the port from 8080 to 8088 and saved the settings.

Now we can access the app from port 8088.



Your server is now running

To replace this page, overwrite the file index.html in the document root folder of this server. The document root folder for this server is the docroot subdirectory of this server's domain directory.

To manage a server on the **local host** with the **default administration port**, go to the Administration Console.

After the server is running, maven is installed in our system using command:

Sudo apt install maven

A maven package is downloaded and extracted in /opt drive.
wget https://www.apache.org/dist/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz -P /tmp
sudo tar xf /tmp/apache-maven-*.tar.gz -C /opt
sudo ln -s /opt/apache-maven-3.6.3 /opt/maven

Then maven.sh file is created
Sudo nano /etc/profile.d/maven.sh
And inside this file, this content is appended:

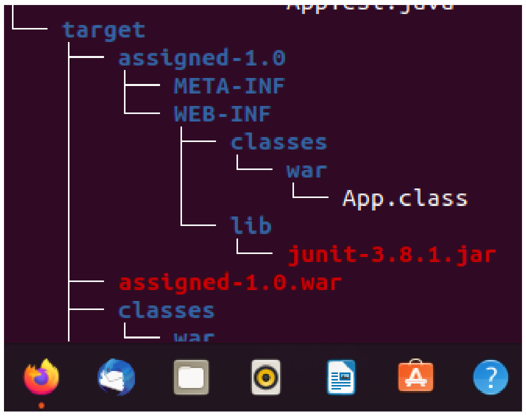
export JAVA_HOME=/usr/lib/jvm/openjdk-11-amd64
export M2_HOME=/opt/maven
export MAVEN_HOME=/opt/maven
export PATH=${M2_HOME}/bin:${PATH}

After that that maven.sh file is made executable and environment variables are loaded using source command as:
sudo chmod +x /etc/profile.d/maven.sh
source /etc/profile.d/maven.sh #environment-variables are loaded

Now maven is initialized in our system.
Now maven is used to generate a sample war file using commands:
Mvn archetype:generate

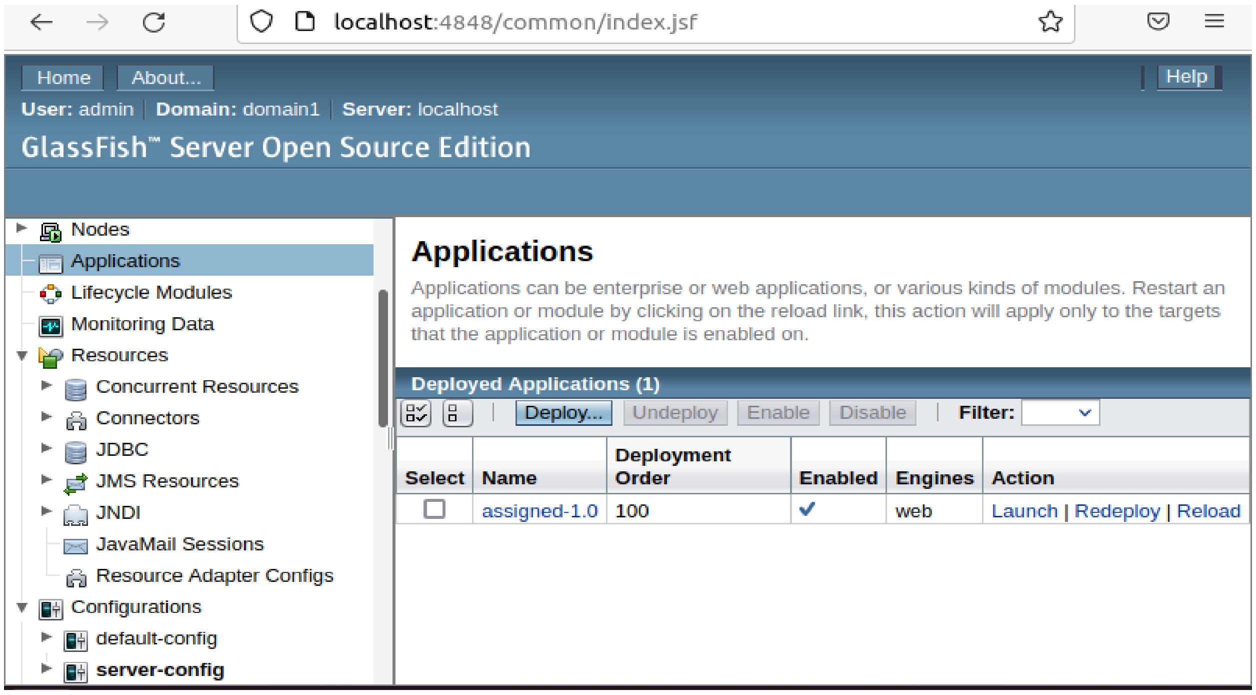
Then go to dir containing pom.xml file and enter
Mvn validate
Mvn compile
Mvn package
Mvn clean install

Then a war file is created in the target directory.



```
│       └── AppTest.java
└── target
    ├── assigned-1.0
    │   ├── META-INF
    │   └── WEB-INF
    │       ├── classes
    │       │   └── war
    │       │       └── App.class
    │       └── lib
    │           └── junit-3.8.1.jar
    ├── assigned-1.0.war
    ├── classes
    │   └── war
```

This war file is then selected from admin server page and launched and deployed.



localhost:4848/common/index.jsf

Home | About... | Help

User: admin | **Domain:** domain1 | **Server:** localhost

GlassFish™ Server Open Source Edition

Nodes
Applications
Lifecycle Modules
Monitoring Data
Resources
Concurrent Resources
Connectors
JDBC
JMS Resources
JNDI
JavaMail Sessions
Resource Adapter Configs
Configurations
default-config
server-config

Applications

Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking on the reload link, this action will apply only to the targets that the application or module is enabled on.

Deployed Applications (1)

Deploy... | Undeploy | Enable | Disable | **Filter:**

Select	Name	Deployment Order	Enabled	Engines	Action
☐	assigned-1.0	100	✓	web	Launch | Redeploy | Reload

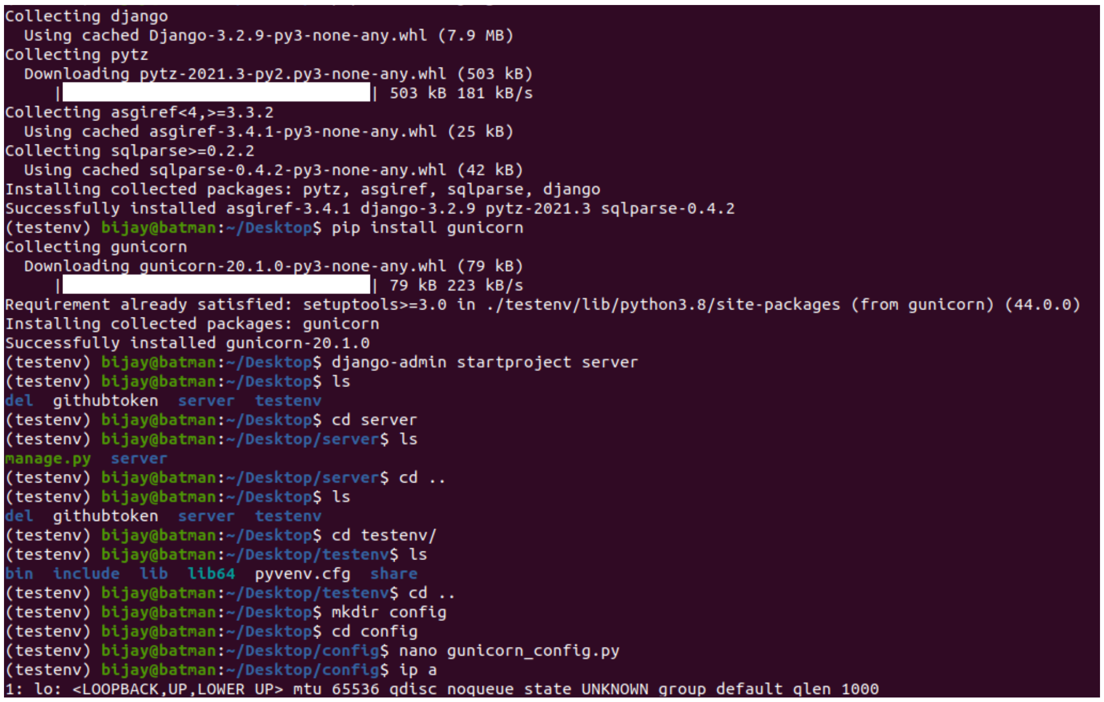
- Create a Django starter project in a separate virtual environment.
- Deploy the 3 instances of application using Gunicorn in 8089 port.
- Dump access log in a file in non-default pattern.
- Dump error log in a file.

The required packages are installed like python3, pip, python3-venv:

Sudo apt install pip
Sudo apt install python3
Sudo apt install python3-venv

Then a virtual environment ‘testenv’ is created
Python3 -m venv testenv
And the virtual environment is activated:
Source testenv/bin/activate
Inside the virtual environment, django and unicorn are installed.
Pip install django
Pip install gunicorn

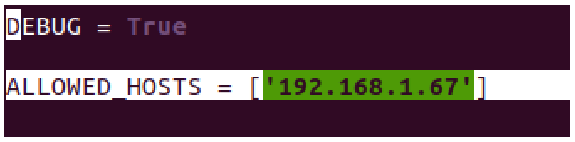
To start a django project named server,
Django-admin startproject server



```
Collecting django
  Using cached Django-3.2.9-py3-none-any.whl (7.9 MB)
Collecting pytz
  Downloading pytz-2021.3-py2.py3-none-any.whl (503 kB)
     |████████████████████████████████| 503 kB 181 kB/s
Collecting asgiref<4,>=3.3.2
  Using cached asgiref-3.4.1-py3-none-any.whl (25 kB)
Collecting sqlparse>=0.2.2
  Using cached sqlparse-0.4.2-py3-none-any.whl (42 kB)
Installing collected packages: pytz, asgiref, sqlparse, django
Successfully installed asgiref-3.4.1 django-3.2.9 pytz-2021.3 sqlparse-0.4.2
(testenv) bijay@batman:~/Desktop$ pip install gunicorn
Collecting gunicorn
  Downloading gunicorn-20.1.0-py3-none-any.whl (79 kB)
     |████████████████████████████████| 79 kB 223 kB/s
Requirement already satisfied: setuptools>=3.0 in ./testenv/lib/python3.8/site-packages (from gunicorn) (44.0.0)
Installing collected packages: gunicorn
Successfully installed gunicorn-20.1.0
(testenv) bijay@batman:~/Desktop$ django-admin startproject server
(testenv) bijay@batman:~/Desktop$ ls
del  githubtoken  server  testenv
(testenv) bijay@batman:~/Desktop$ cd server
(testenv) bijay@batman:~/Desktop/server$ ls
manage.py  server
(testenv) bijay@batman:~/Desktop/server$ cd ..
(testenv) bijay@batman:~/Desktop$ ls
del  githubtoken  server  testenv
(testenv) bijay@batman:~/Desktop$ cd testenv/
(testenv) bijay@batman:~/Desktop/testenv$ ls
bin  include  lib  lib64  pyvenv.cfg  share
(testenv) bijay@batman:~/Desktop/testenv$ cd ..
(testenv) bijay@batman:~/Desktop$ mkdir config
(testenv) bijay@batman:~/Desktop$ cd config
(testenv) bijay@batman:~/Desktop/config$ nano gunicorn_config.py
(testenv) bijay@batman:~/Desktop/config$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
```

Then inside the server project lies a file settings.py. That file is edited to allow hosts:
Sudo nano /server/settings.py

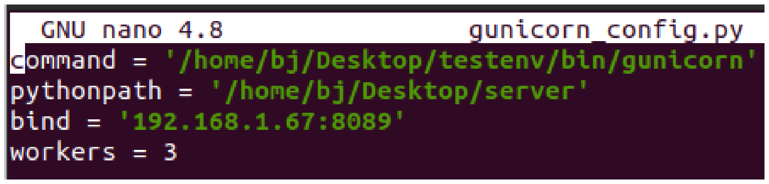
And with allowed hosts, we allow our ip as:
ALLOWED_HOSTS = [‘192.168.1.67’]



```
DEBUG = True

ALLOWED_HOSTS = ['192.168.1.67']
```

A file is created for configuration of gunicorn.
For that, we created a directory and inside it a file gunicorn_config.py.
Nano config/gunicorn_config.py

And inside that file following content is entered:

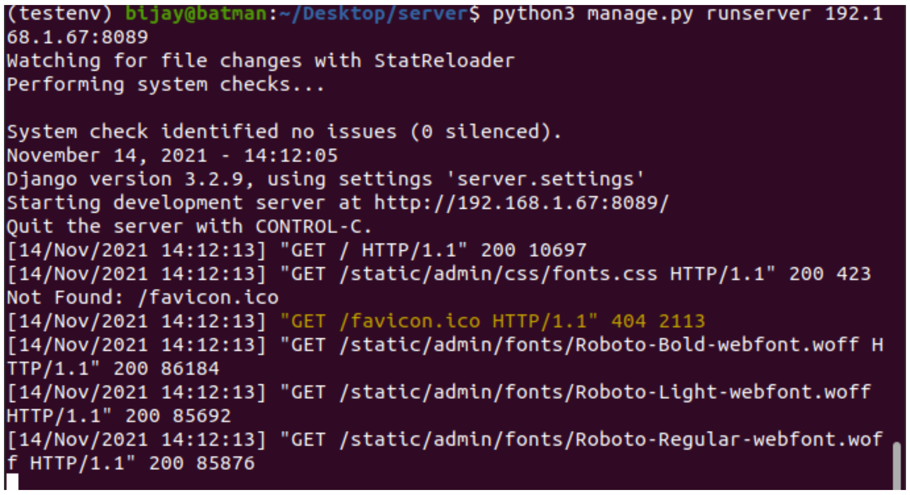


```
  GNU nano 4.8                   gunicorn_config.py
command = '/home/bj/Desktop/testenv/bin/gunicorn'
pythonpath = '/home/bj/Desktop/server'
bind = '192.168.1.67:8089'
workers = 3
```

Now we go to the directory containing manage.py and migrate our settings.
Python3 manage.py makemigrations
Python3 manage.py migrate

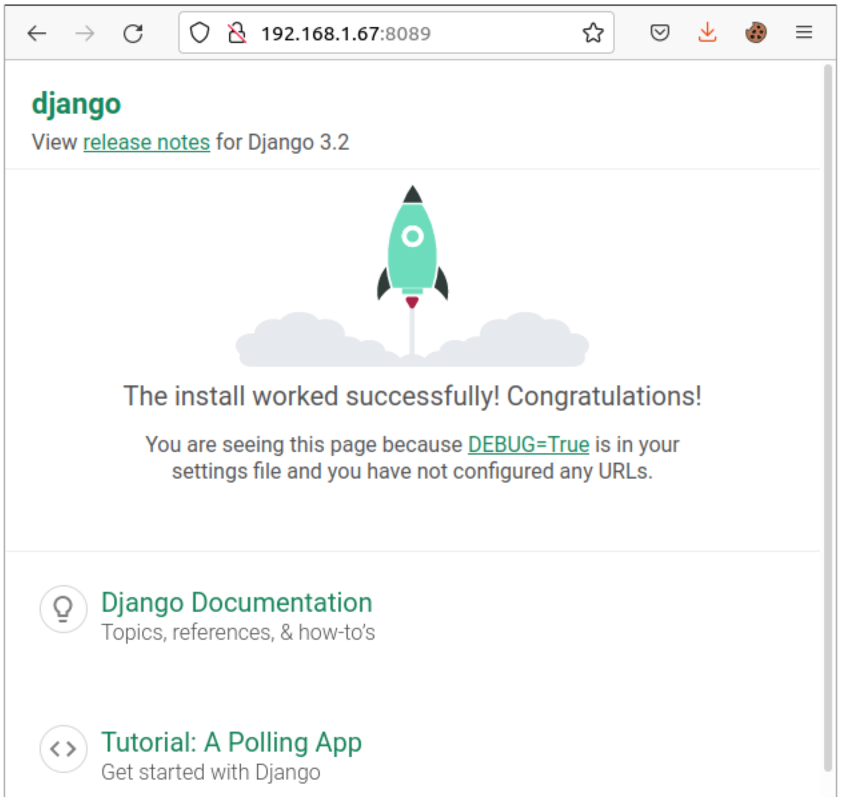
We allow our port in firewall as
Sudo ufw allow 8089

Now when we runserver in this port using command:
Python3 manage.py runserver 192.168.1.67:8089



```
(testenv) bijay@batman:~/Desktop/server$ python3 manage.py runserver 192.1
68.1.67:8089
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
November 14, 2021 - 14:12:05
Django version 3.2.9, using settings 'server.settings'
Starting development server at http://192.168.1.67:8089/
Quit the server with CONTROL-C.
[14/Nov/2021 14:12:13] "GET / HTTP/1.1" 200 10697
[14/Nov/2021 14:12:13] "GET /static/admin/css/fonts.css HTTP/1.1" 200 423
Not Found: /favicon.ico
[14/Nov/2021 14:12:13] "GET /favicon.ico HTTP/1.1" 404 2113
[14/Nov/2021 14:12:13] "GET /static/admin/fonts/Roboto-Bold-webfont.woff H
TTP/1.1" 200 86184
[14/Nov/2021 14:12:13] "GET /static/admin/fonts/Roboto-Light-webfont.woff
HTTP/1.1" 200 85692
[14/Nov/2021 14:12:13] "GET /static/admin/fonts/Roboto-Regular-webfont.wof
f HTTP/1.1" 200 85876
```

On the browser, when we enter the socket address : 192.168.1.67:8089, we get the following result:



django
View release notes for Django 3.2

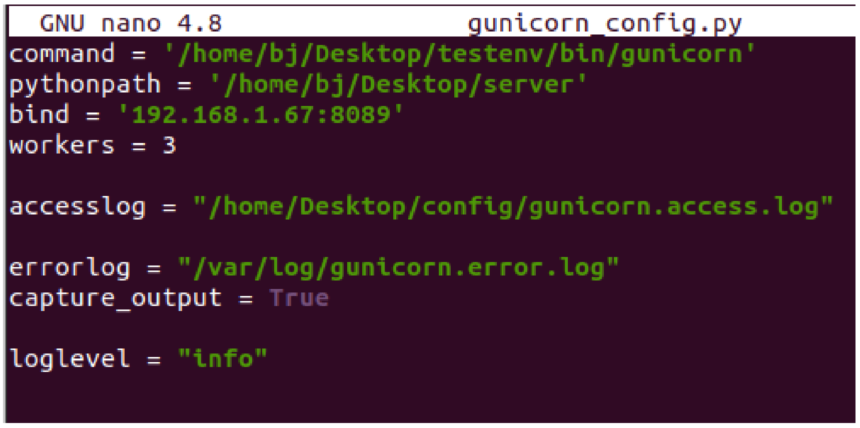
The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

Django Documentation
Topics, references, & how-to's

Tutorial: A Polling App
Get started with Django

For access logs:
The config file of Gunicorn is updated as:
Nano gunicorn_config.py



```
  GNU nano 4.8                      gunicorn_config.py
command = '/home/bj/Desktop/testenv/bin/gunicorn'
pythonpath = '/home/bj/Desktop/server'
bind = '192.168.1.67:8089'
workers = 3

accesslog = "/home/Desktop/config/gunicorn.access.log"

errorlog = "/var/log/gunicorn.error.log"
capture_output = True

loglevel = "info"
```