

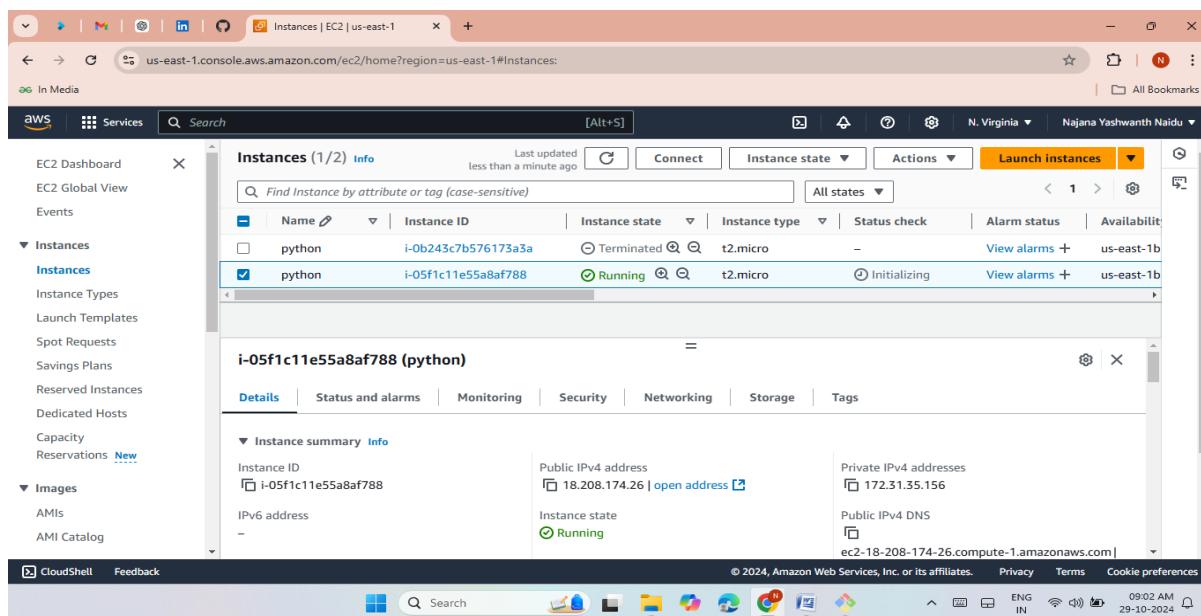
PROJECT

N. Yashwanth Naidu

Method-1

Deploy flask or multiple python web application manually by using aws resources.

- Deploy flask-library-app web application.
- First launch an ec2 instance with the required ports.



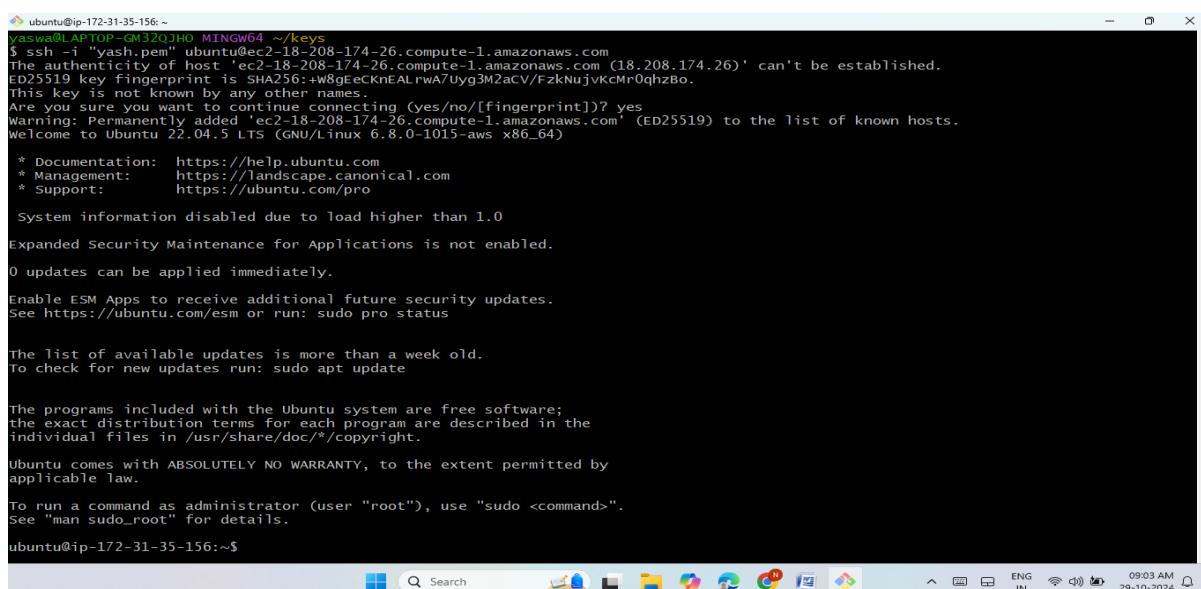
The screenshot shows the AWS Management Console with the EC2 Instances page open. The left sidebar shows various AWS services like EC2 Dashboard, EC2 Global View, Events, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, and AMI Catalog. The main pane shows 'Instances (1/2) info' with two entries:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability zone
python	i-0b243c7b576173a3a	Terminated	t2.micro	-	View alarms +	us-east-1b
python	i-05f1c11e55a8af788	Running	t2.micro	Initializing	View alarms +	us-east-1b

Below the table, a detailed view is shown for the running instance (i-05f1c11e55a8af788). It includes tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. Under Details, the Instance summary section shows:

- Instance ID: i-05f1c11e55a8af788
- Public IPv4 address: 18.208.174.26 [open address]
- Private IPv4 addresses: 172.31.35.156
- Public IPv4 DNS: ec2-18-208-174-26.compute-1.amazonaws.com
- Instance state: Running

- Now connect the server locally.



```
ubuntu@ip-172-31-35-156:~$ ssh -i "yash.pem" ubuntu@ec2-18-208-174-26.compute-1.amazonaws.com
The authenticity of host 'ec2-18-208-174-26.compute-1.amazonaws.com (18.208.174.26)' can't be established.
ED25519 key fingerprint is SHA256:w8gecKnEALrwA7Uyg3M2aCV/FzkNujkvcMr0qhzBo.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-18-208-174-26.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1015-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:   https://landscape.canonical.com
 * Support:      https://ubuntu.com/pro

System information disabled due to load higher than 1.0
Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-35-156:~$
```

➤ Now install python3 and pip3.

```
ubuntu@ip-172-31-35-156:~$ Setting up libjs-sphinxdoc (4.3.2-1) ...
Setting up libgcc-11-dev:amd64 (11.4.0-1ubuntu1~22.04) ...
Setting up gcc-11 (11.4.0-1ubuntu1~22.04) ...
Setting up cpp (4:11.2.0-1ubuntu1) ...
Setting up libc6-dev:amd64 (2.35-0ubuntu3.8) ...
Setting up libtiff5:amd64 (4.3.0-6ubuntu0.10) ...
Setting up libfontconfig1:amd64 (2.13.1-4.2ubuntu5) ...
Setting up gcc (4:11.2.0-1ubuntu1) ...
Setting up libexpat1-dev:amd64 (2.4.7-1ubuntu0.4) ...
Setting up libgd3:amd64 (2.3.0-2ubuntu2) ...
Setting up libstdc++-11-dev:amd64 (11.4.0-1ubuntu1~22.04) ...
Setting up zlib1g-dev:amd64 (1:1.2.11.dfsg-2ubuntu9.2) ...
Setting up libc-devtools (2.35-0ubuntu3.8) ...
Setting up g++-11 (11.4.0-1ubuntu1~22.04) ...
Setting up libpython3.10-dev:amd64 (3.10.12-1~22.04.6) ...
Setting up python3.10-dev (3.10.12-1~22.04.6) ...
Setting up g++ (4:11.2.0-1ubuntu1) ...
update-alternatives: using /usr/bin/g++ to provide /usr/bin/c++ (c++) in auto mode
Setting up build-essential (12.9ubuntu3) ...
Setting up libpython3-dev:amd64 (3.10.6-1~22.04.1) ...
Setting up python3-dev (3.10.6-1~22.04.1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.8) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-35-156:~$ sudo python3 --version
Python 3.10.12
ubuntu@ip-172-31-35-156:~$ |
```

➤ Now install git then clone the flask-library-app.

```
ubuntu@ip-172-31-35-156:~$ Setting up libstdc++-11-dev:amd64 (11.4.0-1ubuntu1~22.04) ...
Setting up zlib1g-dev:amd64 (1:1.2.11.dfsg-2ubuntu9.2) ...
Setting up libc-devtools (2.35-0ubuntu3.8) ...
Setting up g++-11 (11.4.0-1ubuntu1~22.04) ...
Setting up libpython3.10-dev:amd64 (3.10.12-1~22.04.6) ...
Setting up python3.10-dev (3.10.12-1~22.04.6) ...
Setting up g++ (4:11.2.0-1ubuntu1) ...
update-alternatives: using /usr/bin/g++ to provide /usr/bin/c++ (c++) in auto mode
Setting up build-essential (12.9ubuntu3) ...
Setting up libpython3-dev:amd64 (3.10.6-1~22.04.1) ...
Setting up python3-dev (3.10.6-1~22.04.1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.8) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-35-156:~$ sudo python3 --version
Python 3.10.12
ubuntu@ip-172-31-35-156:~$ git clone https://github.com/Yashwanth-najana/flask-library-app.git
Cloning into 'flask-library-app'...
remote: Enumerating objects: 185, done.
remote: Counting objects: 100% (23/23), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 185 (delta 21), reused 17 (delta 17), pack-reused 162 (from 1)
Receiving objects: 100% (185/185), 855.63 KiB | 28.52 MiB/s, done.
Resolving deltas: 100% (109/109), done.
ubuntu@ip-172-31-35-156:~$ ls
Flask-Library-app
ubuntu@ip-172-31-35-156:~$ |
```

- now Install requirements packages with the command.

pip3 install –r requirements.txt

```
ubuntu@ip-172-31-35-156:~/flask-library-app
  Downloading greenlet==1.1.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (155 kB)
Collecting gunicorn==20.1.0
  Downloading gunicorn-20.1.0-py3-none-any.whl (79 kB)
Requirement already satisfied: idna==3.3 in /usr/lib/python3/dist-packages (from -r requirements.txt (line 9)) (3.3)
Collecting itsdangerous==2.0.1
  Downloading itsdangerous-2.0.1-py3-none-any.whl (18 kB)
Collecting Jinja2==3.0.2
  Downloading Jinja2-3.0.2-py3-none-any.whl (133 kB)
Requirement already satisfied: MarkupSafe==2.0.1 in /usr/lib/python3/dist-packages (from -r requirements.txt (line 12)) (2.0.1)
Collecting requests==2.26.0
  Downloading requests-2.26.0-py2.py3-none-any.whl (62 kB)
Collecting SQLAlchemy==1.4.26
  Downloading SQLAlchemy-1.4.26-cp310-cp310-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.6 MB)
Requirement already satisfied: urllib3==1.26.7
  Downloading urllib3-1.26.7-py2.py3-none-any.whl (138 kB)
Collecting Werkzeug==2.0.2
  Downloading Werkzeug-2.0.2-py3-none-any.whl (288 kB)
Requirement already satisfied: setuptools>=3.0 in /usr/lib/python3/dist-packages (from gunicorn==20.1.0->r requirements.txt (line 8))
(59.6.0)
Installing collected packages: Werkzeug, urllib3, Jinja2, itsdangerous, gunicorn, greenlet, charset-normalizer, certifi, SQLAlchemy, requests, Flask, Flask-SQLAlchemy
  WARNING: The script gunicorn is installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
  WARNING: The script normalizer is installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
  WARNING: The script flask is installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed Flask-2.0.2 Flask-SQLAlchemy-2.5.1 Jinja2-3.0.2 SQLAlchemy-1.4.26 werkzeug-2.0.2 certifi-2022.12.7 charset-normalizer-2.0.7 greenlet-1.1.2 gunicorn-20.1.0 itsdangerous-2.0.1 requests-2.26.0 urllib3-1.26.7
ubuntu@ip-172-31-35-156:~/flask-library-app$
```

- Here, we want to edit the file app.py with some details.
 - Because it will generate localhost IP address.
 - We can't access web app with that IP address.

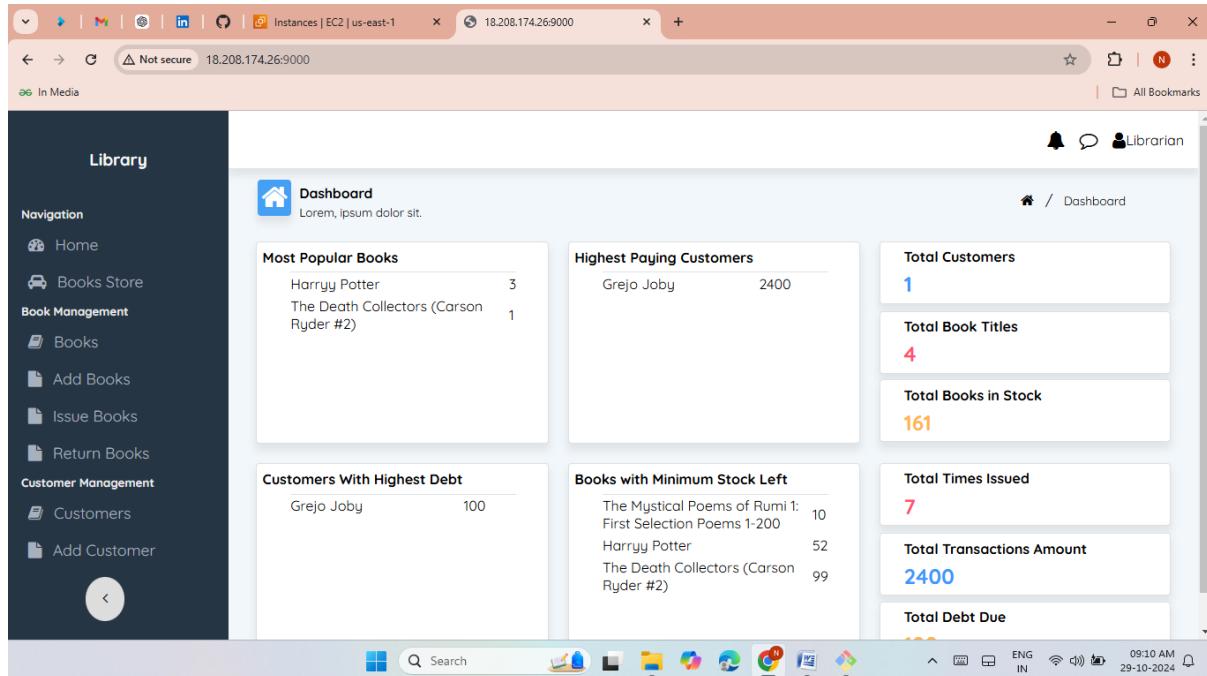
Sudo vi app.py

- Go to very bottom of the file and paste this below txt and save the file.
app.run(host='0.0.0.0', port=8000, debug=True)
 - now run flask-library-app with the command.

python3 app.py or screen -m -d python3 app.py

```
ubuntu@ip-172-31-35-156:~/flask-library-app
* Running on http://172.31.35.156:9000/ (Press CTRL+C to quit)
* Restarting with stat
/home/ubuntu/.local/lib/python3.10/site-packages/flask_sqlalchemy/_init__.py:872: FSADeprecationWarning: SQLALCHEMY_TRACK_MODIFICATION adds significant overhead and will be disabled by default in the future. Set it to True or False to suppress this warning.
    warnings.warn(FSADeprecationWarning(
        * Debugger is active!
        * Debugger PIN: 138-225-829
171.61.102.221 - - [29/oct/2024 03:40:02] code 400, message Bad request version ('jj\x13\x01\x13\x02\x13\x03A+A,A,01@I A\x13A\x14\x00\x9c\x00\x9d\x00\x05\x01\x00\x065')
171.61.102.221 - - [29/oct/2024 03:40:02] "AkçAtıñupö[5,iÖöhicâxö BÄy}éeaÄw:ÄmZçdÄc*6E":ç' jjA+A,A,01@I AA/5S" HTTPStatus.BAD_REQUEST -
171.61.102.221 - - [29/oct/2024 03:40:03] code 400, message Bad request version ('c')
171.61.102.221 - - [29/oct/2024 03:40:03] "aün"Tla_49irYöSc" HTTPStatus.BAD_REQUEST -
171.61.102.221 - - [29/oct/2024 03:40:03] "GET / HTTP/1.1" 200 -
171.61.102.221 - - [29/oct/2024 03:40:03] "GET /static/css/style.css HTTP/1.1" 200 -
171.61.102.221 - - [29/oct/2024 03:40:03] "GET /static/js/script.js HTTP/1.1" 200 -
171.61.102.221 - - [29/oct/2024 03:40:04] "GET /favicon.ico HTTP/1.1" 404 -
^Z
[[1]+ Stopped python3 app.py
ubuntu@ip-172-31-35-156:~/flask-library-app$ screen -m -d python3 app.py
ubuntu@ip-172-31-35-156:~/flask-library-app$ python3 app.py
/home/ubuntu/.local/lib/python3.10/site-packages/flask_sqlalchemy/_init__.py:872: FSADeprecationWarning: SQLALCHEMY_TRACK_MODIFICATION adds significant overhead and will be disabled by default in the future. Set it to True or False to suppress this warning.
    warnings.warn(FSADeprecationWarning(
        * Serving Flask app 'app' (lazy loading)
        * Environment: production
        WARNING: This is a development server. Do not use it in a production deployment.
        Use a production WSGI server instead.
        * Debug mode: on
Traceback (most recent call last):
  File "/home/ubuntu/flask-library-app/app.py", line 268, in <module>
    app.run(host='0.0.0.0', port=9000, debug=True)
  File "/home/ubuntu/.local/lib/python3.10/site-packages/flask/app.py", line 920, in run
    run_simple(t.cast(str, host), port, self, **options)
  File "/home/ubuntu/.local/lib/python3.10/site-packages/werkzeug/serving.py", line 984, in run_simple
    s.bind(server_address)
OSError: [Errno 98] Address already in use
ubuntu@ip-172-31-35-156:~/flask-library-app$
```

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the flask-library-app is displays.



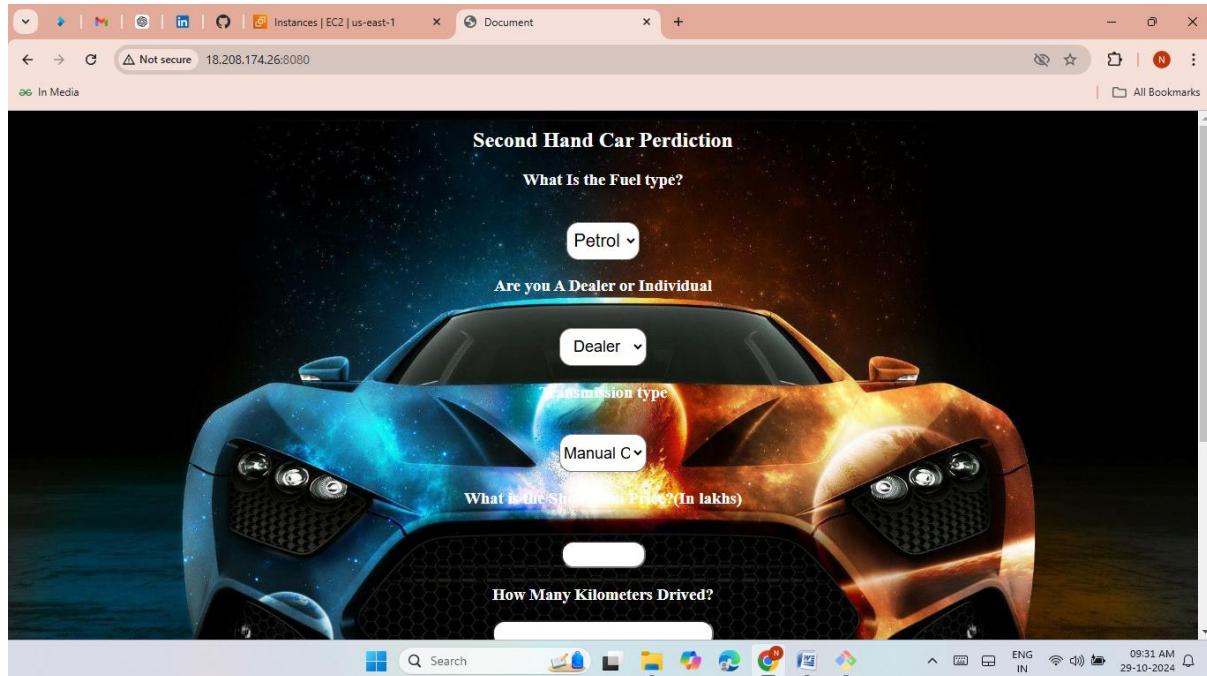
- Now clone the car-prediction python web applications.
- Now install requirements packages with the command.
pip3 install -r requirements.txt
- Now run car-prediction with the command.
python3 app.py or screen -m -d python3 app.py

```
ubuntu@ip-172-31-35-156:~/car-prediction
ubuntu@ip-172-31-35-156:~$ git clone https://github.com/Yashwanth-najana/car-prediction.git
Cloning into 'car-prediction'...
remote: Enumerating objects: 10838, done.
remote: Counting objects: 100% (23/23), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 10838 (delta 9), reused 21 (delta 9), pack-reused 10815 (from 1)
Receiving objects: 100% (10838/10838), 80.95 MiB | 21.90 MiB/s, done.
Resolving deltas: 100% (863/863), done.
Updating files: 100% (10350/10350), done.
ubuntu@ip-172-31-35-156:~$ ls
car-prediction
ubuntu@ip-172-31-35-156:~$ cd car-prediction/
ubuntu@ip-172-31-35-156:~/car-prediction$ ls
OneNotee.joblib  Untitled.ipynb  'car data.csv'          cost                model.pkl      templates
Procfile         app.py        'car prediction deployment link.txt'  gitattributes  requirements.txt
ubuntu@ip-172-31-35-156:~/car-prediction$ |
```

```
ubuntu@ip-172-31-35-156:~/car-prediction
Building wheels for collected packages: sklearn
  Building wheel for sklearn (setup.py) ... done
    Created wheel for sklearn: filename=sklearn-0.0-py2.py3-none-any.whl size=1310 sha256=7dec9e54004964764f522ed0ccf4ea4656cb2306fb0064
b0b9d9084ac034f39b
    Stored in directory: /home/ubuntu/.cache/pip/wheels/9b/13/01/6f3a7fd641f90e1f6c8c7cded057f3394f451f340371c68f3d
Successfully built sklearn
Installing collected packages: pytz, threadpoolctl, python-dateutil, numpy, MarkupSafe, joblib, itsdangerous, gunicorn, colorama, click, Werkzeug, scipy, pandas, Jinja2, scikit-learn, Flask, sklearn
  WARNING: The scripts f2py, f2py3 and F2Py3.10 are installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Attempting uninstall: itsdangerous
  Found existing installation: itsdangerous 2.0.1
  Uninstalling itsdangerous-2.0.1:
    Successfully uninstalled itsdangerous-2.0.1
Attempting uninstall: gunicorn
  Found existing installation: gunicorn 20.1.0
  Uninstalling gunicorn-20.1.0:
    Successfully uninstalled gunicorn-20.1.0
  WARNING: The script gunicorn is installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Attempting uninstall: Werkzeug
  Found existing installation: Werkzeug 2.0.2
  Uninstalling Werkzeug-2.0.2:
    Successfully uninstalled Werkzeug-2.0.2
Attempting uninstall: Jinja2
  Found existing installation: Jinja2 3.0.2
  Uninstalling Jinja2-3.0.2:
    Successfully uninstalled Jinja2-3.0.2
Attempting uninstall: Flask
  Found existing installation: Flask 2.0.2
  Uninstalling Flask-2.0.2:
    Successfully uninstalled Flask-2.0.2
  WARNING: The script flask is installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed Flask-2.2.2 Jinja2-3.1.2 MarkupSafe-2.1.1 werkzeug-2.2.2 click-8.1.3 colorama-0.4.5 gunicorn-20.0.4 itsdangerous-2.1.2 joblib-1.1.0 numpy-1.23.3 pandas-1.4.4 python-dateutil-2.8.2 pytz-2022.2.1 scikit-learn-1.1.2 scipy-1.9.1 sklearn-0.0 threadpoolctl-3.1.0
ubuntu@ip-172-31-35-156:~/car-prediction$ |
```

```
ubuntu@ip-172-31-35-156:~/car-prediction
  Successfully uninstalled Jinja2-3.0.2
Attempting uninstall: Flask
  Found existing installation: Flask 2.0.2
  Uninstalling Flask-2.0.2:
    Successfully uninstalled Flask-2.0.2
  WARNING: The script flask is installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed Flask-2.2.2 Jinja2-3.1.2 MarkupSafe-2.1.1 werkzeug-2.2.2 click-8.1.3 colorama-0.4.5 gunicorn-20.0.4 itsdangerous-2.1.2 joblib-1.1.0 numpy-1.23.3 pandas-1.4.4 python-dateutil-2.8.2 pytz-2022.2.1 scikit-learn-1.1.2 scipy-1.9.1 sklearn-0.0 threadpoolctl-3.1.0
ubuntu@ip-172-31-35-156:~/car-prediction$ screen -m -d python3 app.py
ubuntu@ip-172-31-35-156:~/car-prediction$ python3 app.py
* Serving Flask app 'app'
* Debug mode: on
Address already in use
Port 7000 is in use by another program. Either identify and stop that program, or start the server with a different port.
ubuntu@ip-172-31-35-156:~/car-prediction$ ls
OneHotee.joblib Untitled.ipynb 'car data.csv'          cost           model.pkl      templates
Procfile        app.py      'car prediction deployment link.txt' gitattributes requirements.txt
ubuntu@ip-172-31-35-156:~/car-prediction$ sudo vi app.py
ubuntu@ip-172-31-35-156:~/car-prediction$ python3 app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8080
* Running on http://172.31.35.156:8080
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 138-225-829
172.16.1.102.221 - - [29/Oct/2024 04:01:54] "GET / HTTP/1.1" 200 -
172.16.1.102.221 - - [29/Oct/2024 04:01:56] "GET /favicon.ico HTTP/1.1" 404 -
^Cubuntu@ip-172-31-35-156:~/car-prediction$ screen -m -d python3 app.py
ubuntu@ip-172-31-35-156:~/car-prediction$ ls
OneHotee.joblib Untitled.ipynb 'car data.csv'          cost           model.pkl      templates
Procfile        app.py      'car prediction deployment link.txt' gitattributes requirements.txt
ubuntu@ip-172-31-35-156:~/car-prediction$ |
```

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the car-prediction is displays.



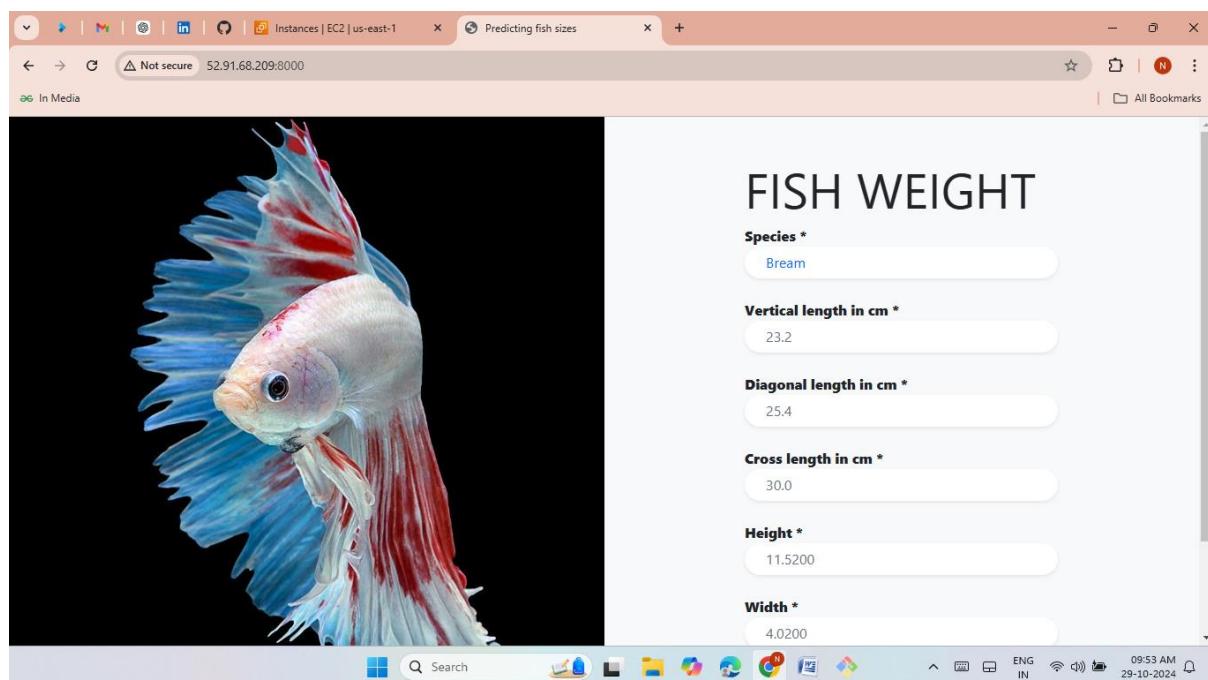
- Now clone the fish python web applications.
- Now install requirements packages with the command.
pip3 install -r requirements.txt
- Now run fish with the command.
python3 app.py or screen -m -d python3 app.py

```
ubuntu@ip-172-31-35-156:~/fish
ubuntu@ip-172-31-35-156:~$ git clone https://github.com/Yashwanth-najana/fish.git
Cloning into 'fish'...
remote: Enumerating objects: 10765, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 10765 (delta 2), reused 6 (delta 1), pack-reused 10755 (from 1)
Receiving objects: 100% (10765/10765), 81.40 MiB | 26.76 MiB/s, done.
Resolving deltas: 100% (804/804), done.
Updating files: 100% (10267/10267), done.
ubuntu@ip-172-31-35-156:~$ ls
fish
ubuntu@ip-172-31-35-156:~/fish$ ls
'Fish market.ipynb'  Fish.csv  Procfile  Random.pkl  app.py  fish  gitattributes  one_joblib  requirements.txt  templates
ubuntu@ip-172-31-35-156:~/fish$ pip3 install -r requirements.txt
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: click==8.1.3 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 1)) (8.1.3)
Requirement already satisfied: colorama==0.4.5 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 2)) (0.4.5)
Requirement already satisfied: Flask==2.2.2 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 3)) (2.2.2)
Collecting importlib-metadata==4.12.0
  Downloading importlib_metadata-4.12.0-py3-none-any.whl (21 kB)
Requirement already satisfied: itsdangerous==2.1.2 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 5)) (2.1.2)
Requirement already satisfied: Jinja2==3.1.2 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 6)) (3.1.2)
Requirement already satisfied: joblib==1.1.0 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 7)) (1.1.0)
Requirement already satisfied: MarkupSafe==2.1.1 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 8)) (2.1.1)
Requirement already satisfied: numpy==1.23.3 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 9)) (1.23.3)
Requirement already satisfied: pandas==1.4.4 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 10)) (1.4.4)
Requirement already satisfied: python-dateutil==2.8.2 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 11)) (2.8.2)
```

```

ubuntu@ip-172-31-35-156:~/fish
0.0)
Requirement already satisfied: threadpoolctl==3.1.0 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 17)) (3.1.0)
Requirement already satisfied: Werkzeug==2.2.2 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 18)) (2.2.2)
Collecting zipp==3.8.1
  Downloading zipp-3.8.1-py3-none-any.whl (5.6 kB)
Requirement already satisfied: gunicorn==20.0.4 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 20)) (20.0.4)
Requirement already satisfied: setuptools>=3.0 in /usr/lib/python3/dist-packages (from gunicorn==20.0.4->-r requirements.txt (line 20)) (59.6.0)
Installing collected packages: zipp, importlib-metadata
Successfully installed importlib-metadata-4.12.0 zipp-3.8.1
ubuntu@ip-172-31-35-156:~/fish$ ls
'fish market.ipynb'  Fish.csv  Procfile  Random.pkl  app.py  fish  gitattributes  one_joblib  requirements.txt  templates
ubuntu@ip-172-31-35-156:~/fish$ sudo vi app.py
ubuntu@ip-172-31-35-156:~/fish$ screen -m -d python3 app.py
ubuntu@ip-172-31-35-156:~/fish$ python3 app.py
/home/ubuntu/.local/lib/python3.10/site-packages/sklearn/base.py:329: UserWarning: Trying to unpickle estimator DecisionTreeRegressor from version 1.1.1 when using version 1.1.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
/home/ubuntu/.local/lib/python3.10/site-packages/sklearn/base.py:329: UserWarning: Trying to unpickle estimator RandomForestRegressor from version 1.1.1 when using version 1.1.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
/home/ubuntu/.local/lib/python3.10/site-packages/sklearn/base.py:329: UserWarning: Trying to unpickle estimator OneHotEncoder from version 1.1.1 when using version 1.1.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
* Serving Flask app 'app'
* Debug mode: on
Address already in use
Port 8000 is in use by another program. Either identify and stop that program, or start the server with a different port.
ubuntu@ip-172-31-35-156:~/fish$
```

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the fish is displays.

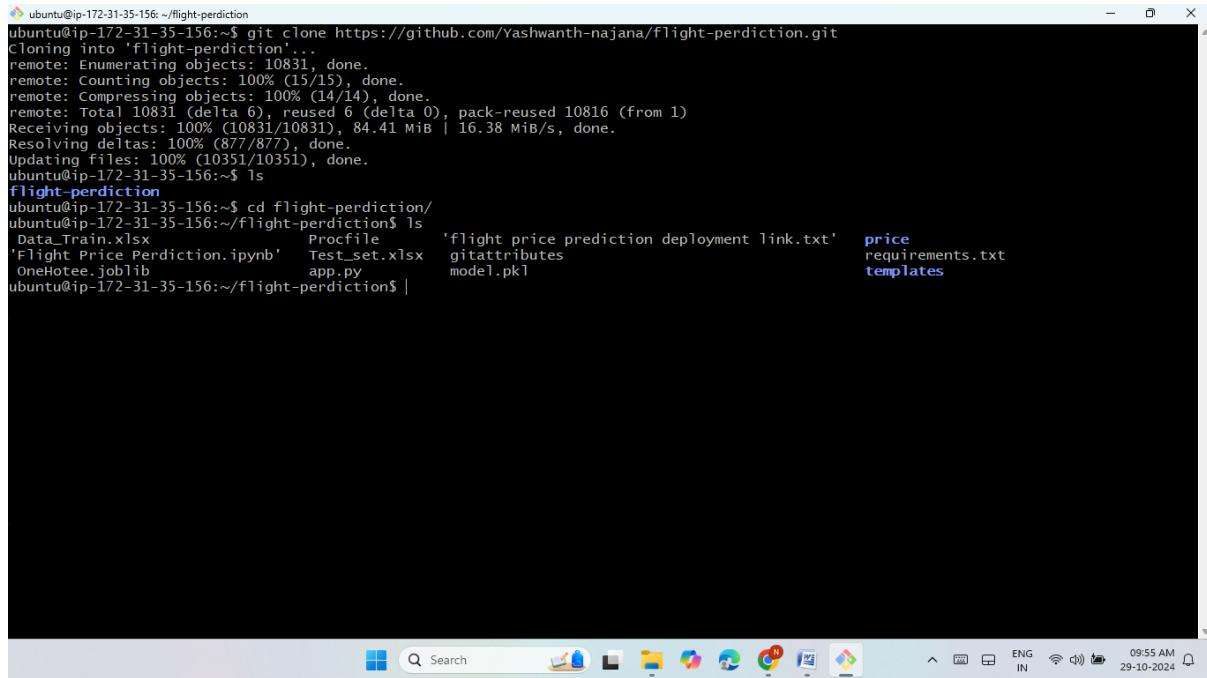


- Now clone the flight-perdiction python web applications.
- Now install requirements packages with the command.

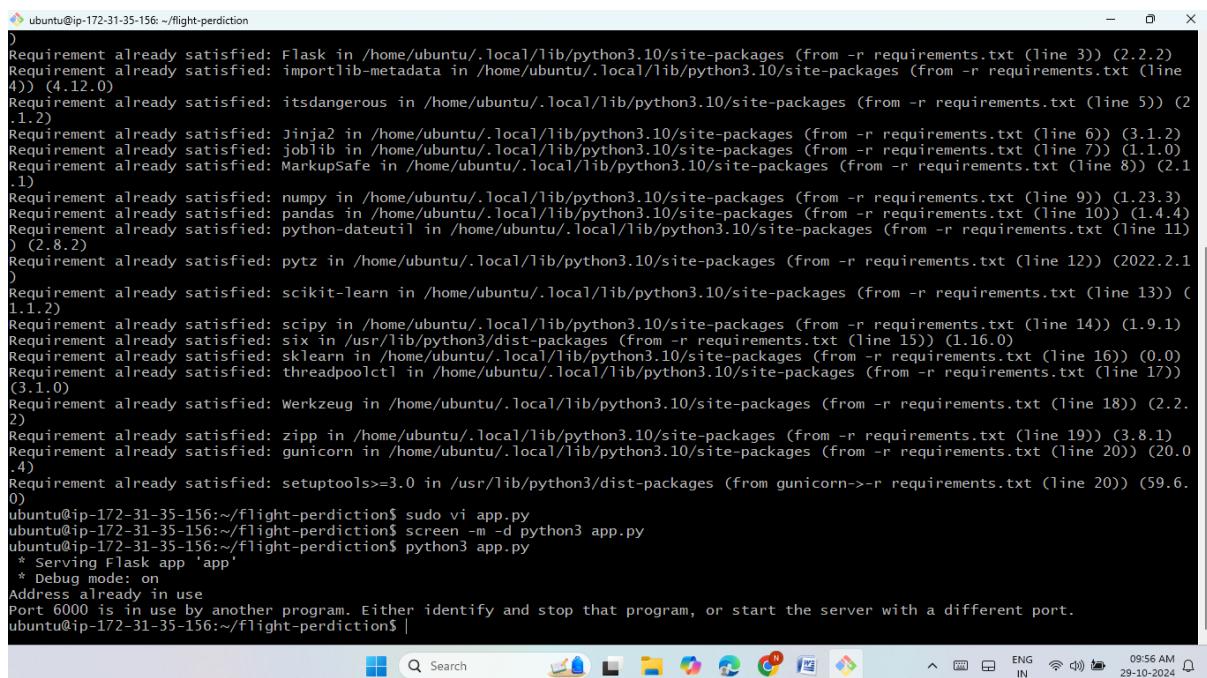
pip3 install -r requirements.txt

- Now run flight-perdiction with the command.

python3 app.py or screen -m -d python3 app.py



```
ubuntu@ip-172-31-35-156:~/flight-perdiction
ubuntu@ip-172-31-35-156:~$ git clone https://github.com/Yashwanth-najana/flight-perdiction.git
Cloning into 'flight-perdiction'...
remote: Enumerating objects: 10831, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 10831 (delta 6), reused 6 (delta 0), pack-reused 10816 (from 1)
Receiving objects: 100% (10831/10831), 84.41 MiB | 16.38 MiB/s, done.
Resolving deltas: 100% (877/877), done.
Updating files: 100% (10351/10351), done.
ubuntu@ip-172-31-35-156:~$ ls
flight-perdiction
ubuntu@ip-172-31-35-156:~$ cd flight-perdiction/
ubuntu@ip-172-31-35-156:~/flight-perdiction$ ls
Data_Train.xlsx      Procfile    'flight price prediction deployment link.txt'  price
'Flight Price Perdiction.ipynb'  Test_set.xlsx  gitattributes  requirements.txt
OneHotee.joblib      app.py     model.pkl   templates
ubuntu@ip-172-31-35-156:~/flight-perdiction$ |
```

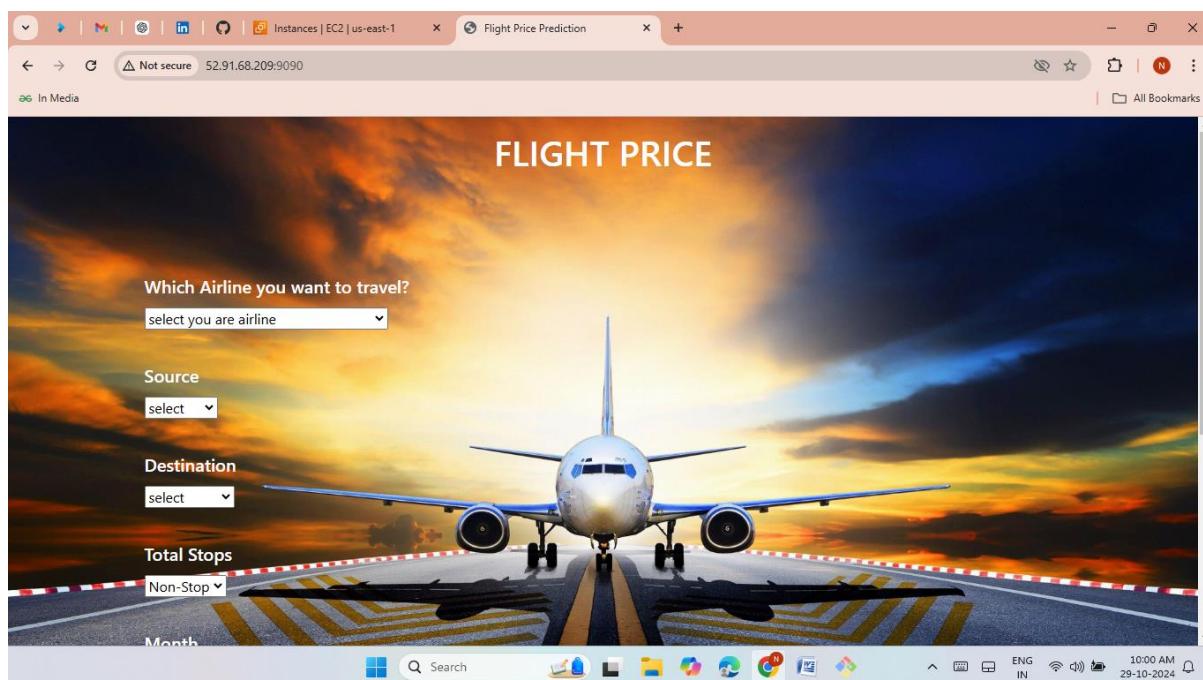


```
Requirement already satisfied: Flask in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 3)) (2.2.2)
Requirement already satisfied: importlib-metadata in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 4)) (4.12.0)
Requirement already satisfied: itsdangerous in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 5)) (2.1.2)
Requirement already satisfied: Jinja2 in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 6)) (3.1.2)
Requirement already satisfied: joblib in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 7)) (1.1.0)
Requirement already satisfied: MarkupSafe in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 8)) (2.1.1)
Requirement already satisfied: numpy in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 9)) (1.23.3)
Requirement already satisfied: pandas in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 10)) (1.4.4)
Requirement already satisfied: python-dateutil in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 11)) (2.8.2)
Requirement already satisfied: pytz in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 12)) (2022.2.1)
Requirement already satisfied: scikit-learn in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 13)) (1.1.2)
Requirement already satisfied: scipy in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 14)) (1.9.1)
Requirement already satisfied: six in /usr/lib/python3/dist-packages (from -r requirements.txt (line 15)) (1.16.0)
Requirement already satisfied: sklearn in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 16)) (0.0)
Requirement already satisfied: threadpoolctl in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 17)) (3.1.0)
Requirement already satisfied: werkzeug in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 18)) (2.2.2)
Requirement already satisfied: zipp in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 19)) (3.8.1)
Requirement already satisfied: gunicorn in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 20)) (20.0.4)
Requirement already satisfied: setuptools>=3.0 in /usr/lib/python3/dist-packages (from gunicorn->-r requirements.txt (line 20)) (59.6.0)
ubuntu@ip-172-31-35-156:~/flight-perdiction$ sudo vi app.py
ubuntu@ip-172-31-35-156:~/flight-perdiction$ screen -m -d python3 app.py
ubuntu@ip-172-31-35-156:~/flight-perdiction$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
Address already in use
Port 6000 is in use by another program. Either identify and stop that program, or start the server with a different port.
ubuntu@ip-172-31-35-156:~/flight-perdiction$ |
```

```

ubuntu@ip-172-31-35-156:~/flight-perdiction
2)
Requirement already satisfied: zipp in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 19)) (3.8.1)
Requirement already satisfied: gunicorn in /home/ubuntu/.local/lib/python3.10/site-packages (from -r requirements.txt (line 20)) (20.0
.).4)
Requirement already satisfied: setuptools>=3.0 in /usr/lib/python3/dist-packages (from gunicorn->-r requirements.txt (line 20)) (59.6.
0)
ubuntu@ip-172-31-35-156:~/flight-perdiction$ sudo vi app.py
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ screen -m -d python3 app.py
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
Address already in use
Port 6000 is in use by another program. Either identify and stop that program, or start the server with a different port.
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
Address already in use
Port 6000 is in use by another program. Either identify and stop that program, or start the server with a different port.
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ sudo vi app.py
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ screen -m -d python3 app.py
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
Address already in use
Port 6000 is in use by another program. Either identify and stop that program, or start the server with a different port.
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ ls
Data_Train.xlsx      Procfile    'flight price prediction deployment link.txt'  price
'Flight Price Perdiction.ipynb' Test_Set.xlsx gitattributes requirements.txt
OneHotee.joblib       app.py     model.pkl templates
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ sudo vi requirements.txt
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ sudo vi app.py
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ screen -m -d python3 app.py
ubuntu@ip-172-31-35-156:~/Flight-perdiction$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
Address already in use
Port 9090 is in use by another program. Either identify and stop that program, or start the server with a different port.
ubuntu@ip-172-31-35-156:~/Flight-perdiction$
```

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the flight-perdiction is displays.



- Now clone the Penguin- python web applications.
- Now install requirements packages with the command.

pip3 install -r requirements.txt

- Now run Penguin- with the command.

python3 app.py or screen -m -d python3 app.py

```
ec2-user@ip-172-31-34-99:~/Penguin- [ec2-user@ip-172-31-34-99 ~]$ git clone https://github.com/Yashwanth-najana/Penguin-.git
Cloning into 'Penguin-'...
remote: Enumerating objects: 10764, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 10764 (delta 3), reused 7 (delta 1), pack-reused 10755 (from 1)
Receiving objects: 100% (10764/10764), 81.28 MiB | 23.97 MiB/s, done.
Resolving deltas: 100% (816/816), done.
Updating files: 100% (10268/10268), done.
[ec2-user@ip-172-31-34-99 ~]$ ls
Penguin-
[ec2-user@ip-172-31-34-99 ~]$ cd Penguin-/
[ec2-user@ip-172-31-34-99 Penguin-]$ ls
app.py      Decision Tree using penguins Data.ipynb  OneHotee.joblib  Procfile      Specie
classifier.pkl  gitattributes          Penguin.ipynb    requirements.txt  templates
[ec2-user@ip-172-31-34-99 Penguin-]$ |
```

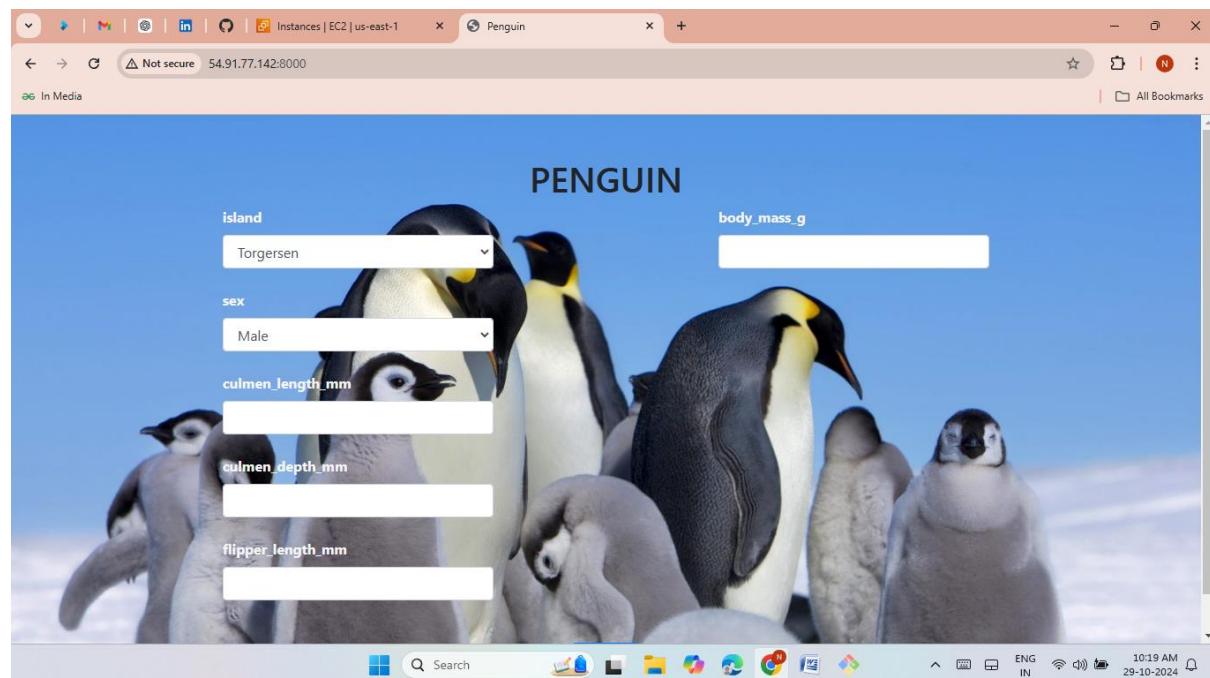
```
ec2-user@ip-172-31-34-99:~/Penguin- [ec2-user@ip-172-31-34-99 ~]$ pip3 install -r requirements.txt
Collecting MarkupSafe-2.1.5-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (25 kB)
  Using cached MarkupSafe-2.1.5-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (25 kB)
Collecting numpy
  Using cached numpy-1.21.6-cp37-cp37m-manylinux_2_12_x86_64.manylinux2010_x86_64.whl (15.7 MB)
Collecting pandas
  Using cached pandas-1.3.5-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.3 MB)
Collecting python-dateutil
  Using cached python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
Collecting pytz
  Using cached pytz-2024.2-py2.py3-none-any.whl (508 kB)
Collecting scikit-learn
  Using cached scikit_learn-1.0.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (24.8 MB)
Collecting scipy
  Using cached scipy-1.7.3-cp37-cp37m-manylinux_2_12_x86_64.manylinux2010_x86_64.whl (38.1 MB)
Collecting six
  Using cached six-1.16.0-py2.py3-none-any.whl (11 kB)
Collecting threadpoolctl
  Downloading threadpoolctl-3.1.0-py3-none-any.whl (14 kB)
Collecting Werkzeug
  Downloading Werkzeug-2.2.3-py3-none-any.whl (233 kB)
    |██████████| 233 kB 25.5 MB/s
Collecting zipp
  Downloading zipp-3.15.0-py3-none-any.whl (6.8 kB)
Collecting gunicorn
  Downloading gunicorn-23.0.0-py3-none-any.whl (85 kB)
    |██████████| 85 kB 7.9 MB/s
Collecting typing_extensions>=3.6.4; python_version < "3.8"
  Downloading typing_extensions-4.7.1-py3-none-any.whl (33 kB)
Collecting packaging
  Downloading packaging-24.0-py3-none-any.whl (53 kB)
    |██████████| 53 kB 6.1 MB/s
Installing collected packages: typing_extensions, zipp, importlib-metadata, click, colorama, itsdangerous, MarkupSafe, Werkzeug, Jinja2, Flask, joblib, numpy, pytz, six, python-dateutil, pandas, threadpoolctl, scipy, scikit-learn, packaging, gunicorn
Successfully installed Flask-2.2.5 Jinja2-3.1.4 MarkupSafe-2.1.5 Werkzeug-2.2.3 click-8.1.7 colorama-0.4.6 gunicorn-23.0.0 importlib-metadata-6.7.0 itsdangerous-2.1.2 joblib-1.3.2 numpy-1.21.6 packaging-24.0 pandas-1.3.5 python-dateutil-2.9.0.post0 pytz-2024.2 scikit-learn-1.0.2 scipy-1.7.3 six-1.16.0 threadpoolctl-3.1.0 typing_extensions-4.7.1 zipp-3.15.0
[ec2-user@ip-172-31-34-99 Penguin-]$ |
```

```

ec2-user@ip-172-31-34-99:~/Penguin-
Collecting zipp
  Downloading zipp-3.15.0-py3-none-any.whl (6.8 kB)
Collecting gunicorn
  Downloading gunicorn-23.0.0-py3-none-any.whl (85 kB)
    ||██████████| 85 kB 7.9 MB/s
collecting typing-extensions>=3.6.4; python_version < "3.8"
  Downloading typing_extensions-4.7.1-py3-none-any.whl (33 kB)
Collecting packaging
  Downloading packaging-24.0-py3-none-any.whl (53 kB)
    ||██████████| 53 kB 6.1 MB/s
Installing collected packages: typing-extensions, zipp, importlib-metadata, click, colorama, itsdangerous, MarkupSafe, Werkzeug, Jinja2, Flask, joblib, numpy, pytz, six, python-dateutil, pandas, threadpoolctl, scipy, scikit-learn, packaging, gunicorn
Successfully installed Flask-2.2.5 Jinja2-3.1.4 MarkupSafe-2.2.3 Werkzeug-2.2.3 click-8.1.7 colorama-0.4.6 gunicorn-23.0.0 importlib-metadata-6.7.0 itsdangerous-2.1.2 joblib-1.3.2 numpy-1.21.6 packaging-24.0 pandas-1.3.5 python-dateutil-2.9.0.post0 pytz-2024.2 scikit-learn-1.0.2 scipy-1.7.3 six-1.16.0 threadpoolctl-3.1.0 typing-extensions-4.7.1 zipp-3.15.0
[ec2-user@ip-172-31-34-99 Penguin]$ ls
app.py      Decision Tree using penguins Data.ipynb OneHotEncoder.joblib Procfile      Specie
classifier.pkl gitattributes Penguin.ipynb requirements.txt templates
[ec2-user@ip-172-31-34-99 Penguin]$ sudo vi app.py
[ec2-user@ip-172-31-34-99 Penguin]$ screen -m -d python3 app.py
[ec2-user@ip-172-31-34-99 Penguin]$ python3 app.py
/home/ec2-user/.local/lib/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle estimator DecisionTreeClassifier from version 1.1.2 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
  UserWarning,
/home/ec2-user/.local/lib/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle estimator OneHotEncoder from version 1.1.2 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
  UserWarning,
  * Serving Flask app 'app'
  * Debug mode: on
Address already in use
Port 8000 is in use by another program. Either identify and stop that program, or start the server with a different port.
[ec2-user@ip-172-31-34-99 Penguin]$

```

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the Penguin- is displays.

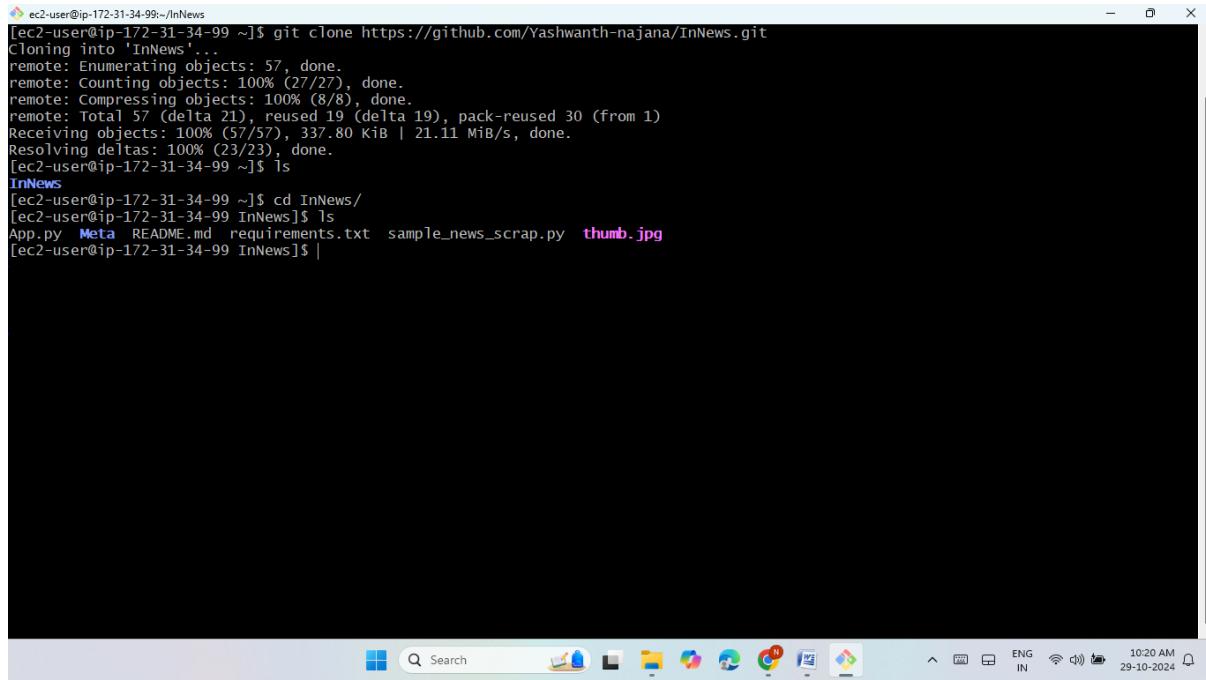


- Now clone the InNews python web applications.
- Now install requirements packages with the command.

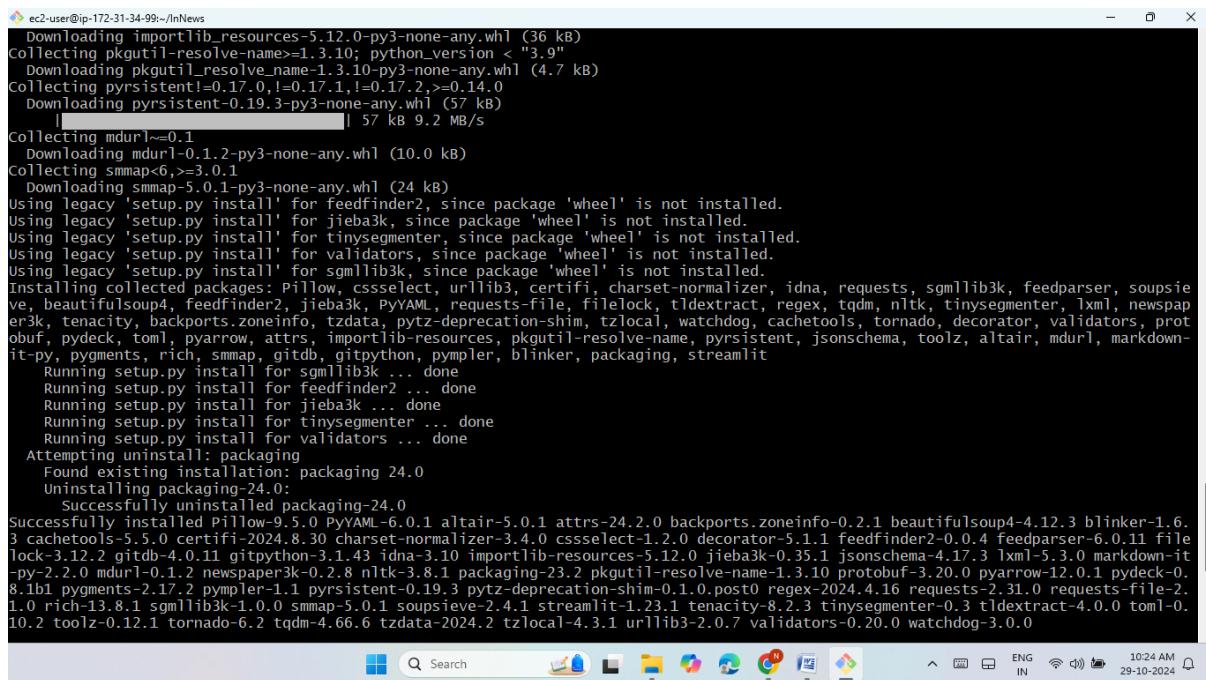
pip3 install -r requirements.txt

- Now run InNews with the command.

python3 app.py or screen -m -d python3 app.py



```
ec2-user@ip-172-31-34-99:~/InNews
[ec2-user@ip-172-31-34-99 ~]$ git clone https://github.com/Yashwanth-najana/InNews.git
Cloning into 'InNews'...
remote: Enumerating objects: 57, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 57 (delta 21), reused 19 (delta 19), pack-reused 30 (from 1)
Receiving objects: 100% (57/57), 337.80 KiB | 21.11 MiB/s, done.
Resolving deltas: 100% (23/23), done.
[ec2-user@ip-172-31-34-99 ~]$ ls
InNews
[ec2-user@ip-172-31-34-99 ~]$ cd InNews/
[ec2-user@ip-172-31-34-99 InNews]$ ls
App.py  Meta  README.md  requirements.txt  sample_news_scrap.py  thumb.jpg
[ec2-user@ip-172-31-34-99 InNews]$ |
```



```
ec2-user@ip-172-31-34-99:~/InNews
Downloading importlib_resources-5.12.0-py3-none-any.whl (36 kB)
Collecting pkgutil_resolve_name==1.3.10; python_version < "3.9"
  Downloading pkgutil_resolve_name-1.3.10-py3-none-any.whl (4.7 kB)
Collecting pyrsistent!=0.17.0,!0.17.1,!0.17.2,>=0.14.0
  Downloading pyrsistent-0.19.3-py3-none-any.whl (57 kB)
    |██████████| 57 kB 9.2 MB/s
Collecting mdurl==0.1
  Downloading mdurl-0.1.2-py3-none-any.whl (10.0 kB)
Collecting smmap<6,>=3.1
  Downloading smmap-5.0.1-py3-none-any.whl (24 kB)
Using legacy 'setup.py install' for feedfinder2, since package 'wheel' is not installed.
Using legacy 'setup.py install' for jieba3k, since package 'wheel' is not installed.
Using legacy 'setup.py install' for tinysegmenter, since package 'wheel' is not installed.
Using legacy 'setup.py install' for validators, since package 'wheel' is not installed.
Using legacy 'setup.py install' for sgmlib3k, since package 'wheel' is not installed.
Installing collected packages: Pillow, cssselect, urllib3, certifi, charset-normalizer, idna, requests, sgmlib3k, feedparser, soupsieve, beautifulsoup4, feedfinder2, jieba3k, PyYAML, requests-file, filelock, tldextract, regex, tqdm, nltk, tinysegmenter, lxml, newspaper3k, tenacity, backports.zoneinfo, tzdata, pytz-deprecation-shim, tzlocal, watchdog, cachetools, tornado, decorator, validators, protobuf, pydeck, toml, pyarrow, attrs, importlib-resources, pkgutil-resolve-name, pyrsistent, jsonschema, toolz, altair, mdurl, markdown-it-py, pygments, rich, smmap, gitdb, gitpython, pympler, blinker, packaging, streamlit
  Running setup.py install for sgmlib3k ... done
  Running setup.py install for feedfinder2 ... done
  Running setup.py install for jieba3k ... done
  Running setup.py install for tinysegmenter ... done
  Running setup.py install for validators ... done
Attempting uninstall: packaging
Found existing installation: packaging 24.0
Uninstalling packaging-24.0:
  Successfully uninstalled packaging-24.0
Successfully installed Pillow-9.5.0 PyYAML-6.0.1 altair-5.0.1 attrs-24.2.0 backports.zoneinfo-0.2.1 beautifulsoup4-4.12.3 blinker-1.6.3 cachetools-5.5.0 certifi-2024.8.30 charset-normalizer-3.4.0 cssselect-1.2.0 decorator-5.1.1 feedfinder2-0.0.4 feedparser-6.0.11 filelock-3.12.2 gitdb-4.0.11 gitpython-3.1.43 idna-3.10 importlib-resources-5.12.0 jieba3k-0.35.1 jsonschema-4.17.3 lxml-5.3.0 markdown-it-py-2.2.0 mdurl-0.1.2 newspaper3k-0.2.8 nltk-3.8.1 packaging-23.2 pkgutil-resolve-name-1.3.10 protobuf-3.20.0 pyarrow-12.0.1 pydeck-0.8.1b1 pygments-2.17.2 pympler-1.1 pyrsistent-0.19.3 pytz-deprecation-shim-0.1.0.post0 regex-2024.4.16 requests-2.31.0 requests-file-2.1.0 rich-13.8.1 sgmlib3k-1.0.0 smmap-5.0.1 soupsieve-2.4.1 streamlit-1.23.1 tenacity-8.2.3 tinysegmenter-0.3 tldextract-4.0.0 toml-0.10.2 toolz-0.12.1 tornado-6.2 tqdm-4.66.6 tzdata-2024.2 tzlocal-4.3.1 urllib3-2.0.7 validators-0.20.0 watchdog-3.0.0
```

```

ec2-user@ip-172-31-34-99:~/InNews
[ec2-user@ip-172-31-34-99 InNews]$ sudo vi App.py
[ec2-user@ip-172-31-34-99 InNews]$ python3 App.py
[nltk_data] Downloading package punkt to /home/ec2-user/nltk_data...
[nltk_data] Package punkt is already up-to-date!
2024-10-29 05:12:02.916
Warning: to view this Streamlit app on a browser, run it with the following
command:

streamlit run App.py [ARGUMENTS]
Traceback (most recent call last):
  File "App.py", line 137, in <module>
    app.run(host='0.0.0.0', port=8080, debug=True)
NameError: name 'app' is not defined
[ec2-user@ip-172-31-34-99 InNews]$ streamlit run App.py
collecting usage statistics. To deactivate, set browser.gatherUsageStats to False.

You can now view your Streamlit app in your browser.

Network URL: http://172.31.34.99:8501
External URL: http://54.91.77.142:8501

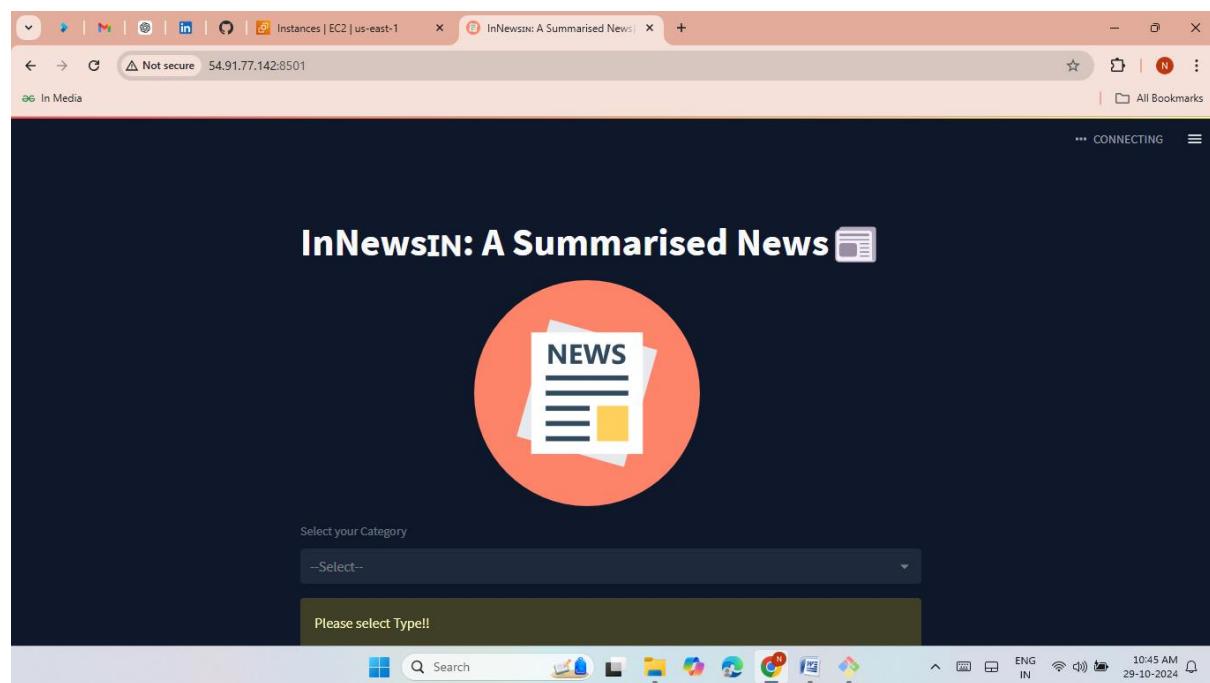
AC Stopping...
[ec2-user@ip-172-31-34-99 InNews]$ sudo vi App.py
[ec2-user@ip-172-31-34-99 InNews]$ screen -m -d python3 App.py
[ec2-user@ip-172-31-34-99 InNews]$ streamlit run App.py
Collecting usage statistics. To deactivate, set browser.gatherUsageStats to False.

You can now view your Streamlit app in your browser.

Network URL: http://172.31.34.99:8501
External URL: http://54.91.77.142:8501

```

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the InNews is displays.



- Now clone the Medical-Insurance python web applications.
- Now install requirements packages with the command.

pip3 install -r requirements.txt

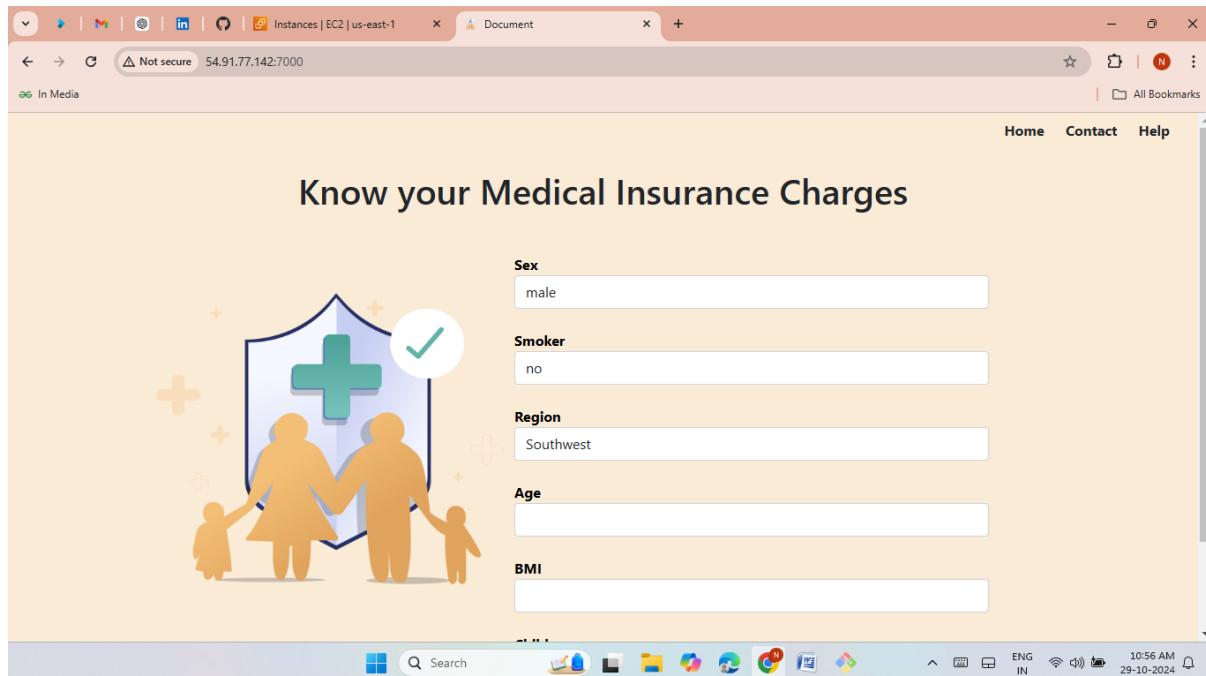
- Now run Medical-Insurance with the command.

python3 app.py or screen -m -d python3 app.py

```
ec2-user@ip-172-31-34-99:~/Medical-Insurance
[ec2-user@ip-172-31-34-99 ~]$ git clone https://github.com/Yashwanth-najana/Medical-Insurance.git
Cloning into 'Medical-Insurance'...
remote: Enumerating objects: 10759, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 10759 (delta 0), reused 0 (delta 0), pack-reused 10755 (from 1)
Receiving objects: 100% (10759/10759), 81.69 MiB | 24.50 MiB/s, done.
Resolving deltas: 100% (818/818), done.
Updating files: 100% (10267/10267), done.
[ec2-user@ip-172-31-34-99 ~]$ ls
Medical-Insurance
[ec2-user@ip-172-31-34-99 ~]$ cd Medical-Insurance/
[ec2-user@ip-172-31-34-99 Medical-Insurance]$ ls
app.py gitattributes Medical Medical Insurance.ipynb OneHotEncoder.joblib Procfile regressor.pkl requirements.txt templates
[ec2-user@ip-172-31-34-99 Medical-Insurance]$
```

```
Requirement already satisfied: threadpoolctl in /home/ec2-user/.local/lib/python3.7/site-packages (from -r requirements.txt (line 16)) (3.1.0)
Requirement already satisfied: werkzeug in /home/ec2-user/.local/lib/python3.7/site-packages (from -r requirements.txt (line 17)) (2.2.3)
Requirement already satisfied: zipp in /home/ec2-user/.local/lib/python3.7/site-packages (from -r requirements.txt (line 18)) (3.15.0)
Requirement already satisfied: gunicorn in /home/ec2-user/.local/lib/python3.7/site-packages (from -r requirements.txt (line 19)) (23.0.0)
Requirement already satisfied: typing-extensions>=3.6.4; python_version < "3.8" in /home/ec2-user/.local/lib/python3.7/site-packages (from importlib-metadata->-r requirements.txt (line 4)) (4.7.1)
Requirement already satisfied: packaging in /home/ec2-user/.local/lib/python3.7/site-packages (from gunicorn->-r requirements.txt (line 19)) (23.2)
[ec2-user@ip-172-31-34-99 Medical-Insurance]$ sudo vi app.py
[ec2-user@ip-172-31-34-99 Medical-Insurance]$ screen -m -d python3 app.py
[ec2-user@ip-172-31-34-99 Medical-Insurance]$ python3 app.py
:bash: python3: command not found
[ec2-user@ip-172-31-34-99 Medical-Insurance]$ python3 app.py
/home/ec2-user/.local/lib/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle estimator DecisionTreeRegressor from version 1.1.2 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning,
/home/ec2-user/.local/lib/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle estimator RandomForestRegressor from version 1.1.2 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning,
/home/ec2-user/.local/lib/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle estimator OneHotEncoder from version 1.1.2 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning,
* Serving Flask app 'app'
* Debug mode: on
Address already in use
Port 7000 is in use by another program. Either identify and stop that program, or start the server with a different port.
[ec2-user@ip-172-31-34-99 Medical-Insurance]$ |
```

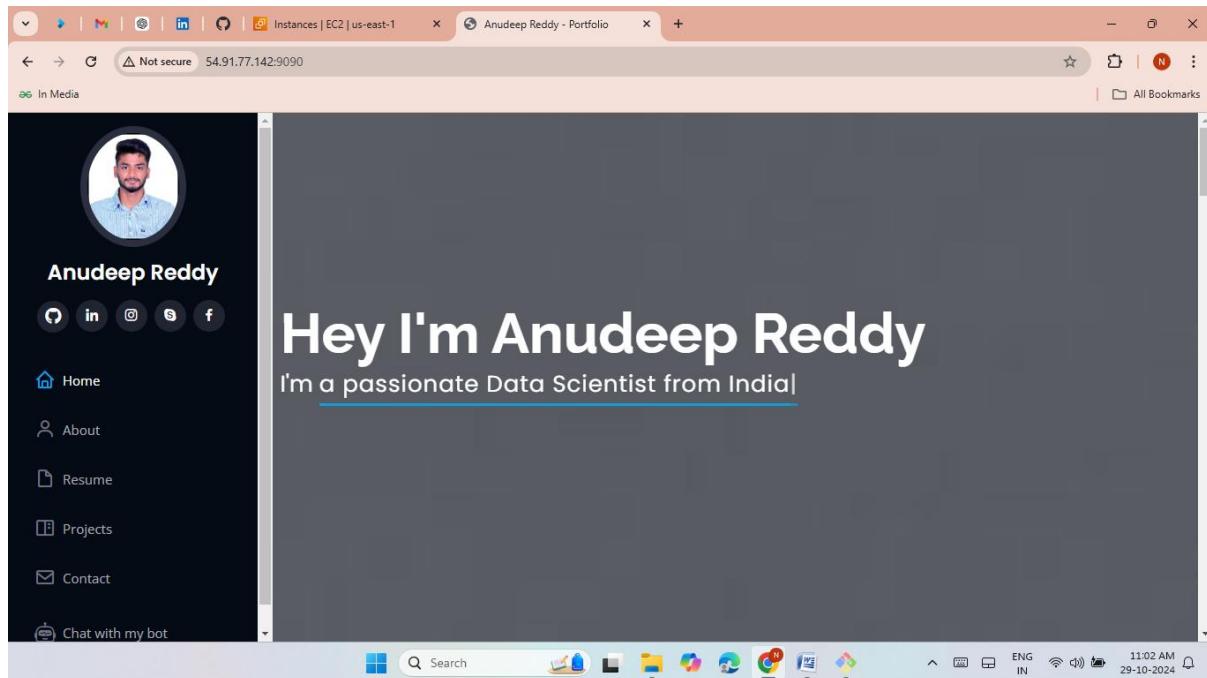
- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the Medical-Insurance is displays.



- Now clone the Portfolio python web applications.
- Now install requirements packages with the command.
pip3 install -r requirements.txt
- Now run Portfolio with the command.
python3 app.py or screen -m -d python3 app.py

```
ec2-user@ip-172-31-34-99:~/Portfolio$ git clone https://github.com/Yashwanth-najana/Portfolio.git
Cloning into 'Portfolio'...
remote: Enumerating objects: 1971, done.
remote: Counting objects: 100% (1971/1971), done.
remote: Compressing objects: 100% (1794/1794), done.
remote: Total 1971 (delta 149), reused 1966 (delta 147), pack-reused 0 (from 0)
Receiving objects: 100% (1971/1971), 8.74 MiB | 24.58 MiB/s, done.
Resolving deltas: 100% (149/149), done.
[ec2-user@ip-172-31-34-99 ~]$ ls
Portfolio
[ec2-user@ip-172-31-34-99 ~]$ cd Portfolio/
[ec2-user@ip-172-31-34-99 Portfolio]$ LS
-bash: LS: command not found
[ec2-user@ip-172-31-34-99 Portfolio]$ ls
app.py gitattributes MyPortfolio Procfile requirements.txt static templates
[ec2-user@ip-172-31-34-99 Portfolio]$ |
```

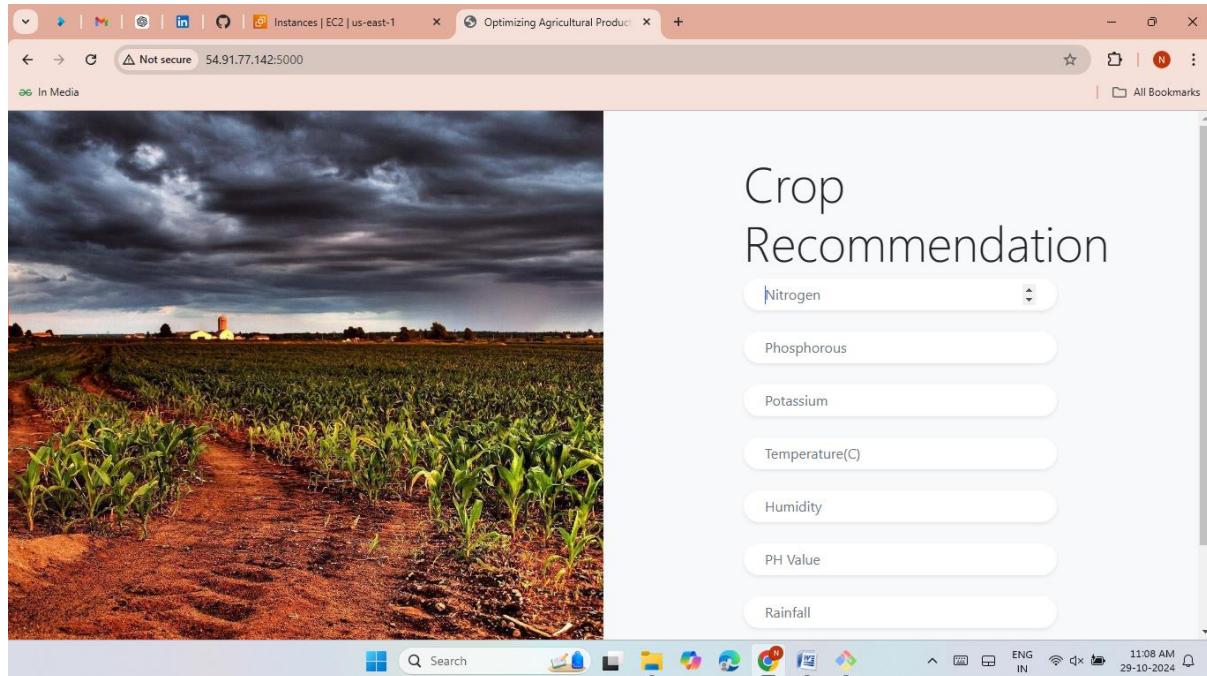
- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the Portfolio is displays.



- Now clone the Agri python web applications.
- Now install requirements packages with the command.
pip3 install -r requirements.txt
- Now run Agri with the command.
python3 app.py or screen -m -d python3 app.py

```
ec2-user@ip-172-31-34-99:~/Agri
[ec2-user@ip-172-31-34-99 ~]$ git clone https://github.com/Yashwanth-najana/Agri.git
Cloning into 'Agri'...
remote: Enumerating objects: 10765, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 10765 (delta 3), reused 6 (delta 2), pack-reused 10755 (from 1)
Receiving objects: 100% (10765/10765), 81.97 MiB | 24.73 MiB/s, done.
Resolving deltas: 100% (813/813), done.
Updating files: 100% (10267/10267), done.
[ec2-user@ip-172-31-34-99 ~]$ lsls
-bash: lsls: command not found
[ec2-user@ip-172-31-34-99 ~]$ ls
Agri
[ec2-user@ip-172-31-34-99 ~]$ cd Agri/
[ec2-user@ip-172-31-34-99 Agri]$ ls
Agri Agriculturedeploy - Shortcut.lnk app.py gitattributes requirements.txt
Agriculture.csv Agriculture.ipynb decmodel.pkl Procfile templates
[ec2-user@ip-172-31-34-99 Agri]$ |
```

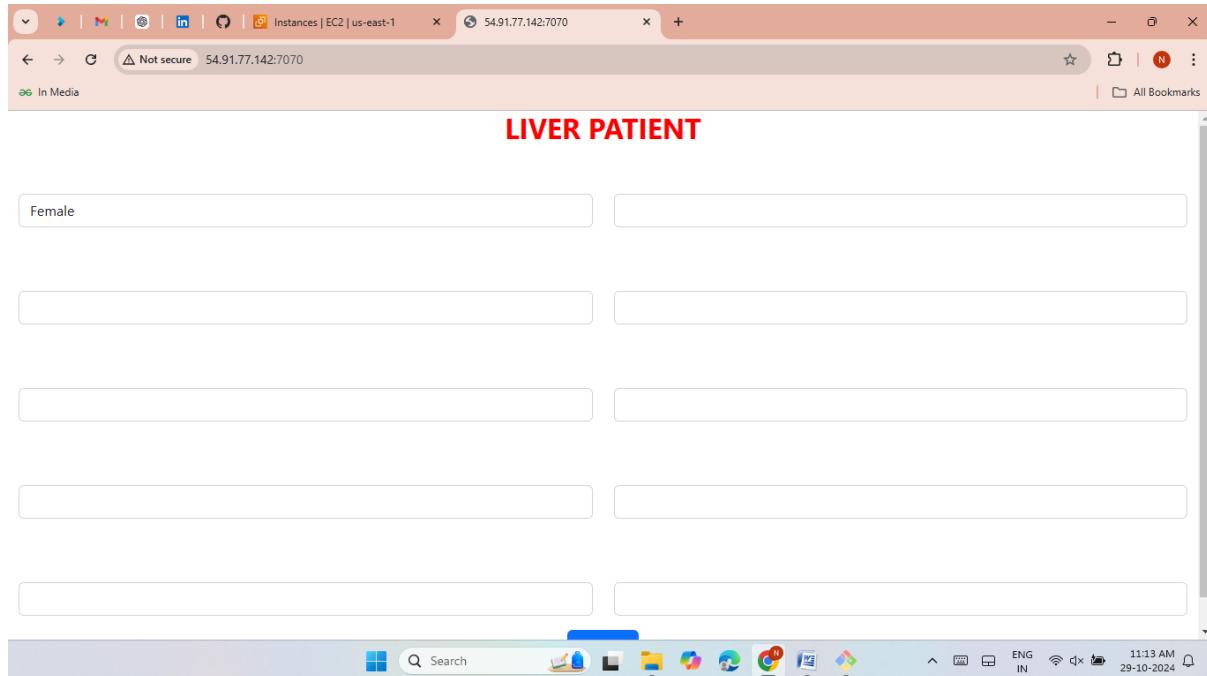
- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the Agri is displays.



- Now clone the Indian-liver-patients python web applications.
- Now install requirements packages with the command.
pip3 install -r requirements.txt
- Now run Indian-liver-patients with the command.
python3 app.py or screen -m -d python3 app.py

```
ec2-user@ip-172-31-34-99:~/indian-liver-patients$ git clone https://github.com/Yashwanth-najana/indian-liver-patients.git
Cloning into 'indian-liver-patients'...
remote: Enumerating objects: 10757, done.
remote: Total 10757 (delta 0), reused 0 (delta 0), pack-reused 10757 (from 1)
Receiving objects: 100% (10757/10757), 83.42 MiB | 24.76 MiB/s, done.
Resolving deltas: 100% (815/815), done.
Updating files: 100% (10268/10268), done.
[ec2-user@ip-172-31-34-99 ~]$ ls
indian-liver-patients
[ec2-user@ip-172-31-34-99 ~]$ cd indian-liver-patients/
[ec2-user@ip-172-31-34-99 indian-liver-patients]$ ls
app.py      Indian Liver Patient.csv  liver      ohe_joblib  requirements.txt
gitattributes  indian liver patient.ipynb  logmodel.pkl  Procfile  templates
[ec2-user@ip-172-31-34-99 indian-liver-patients]$ |
```

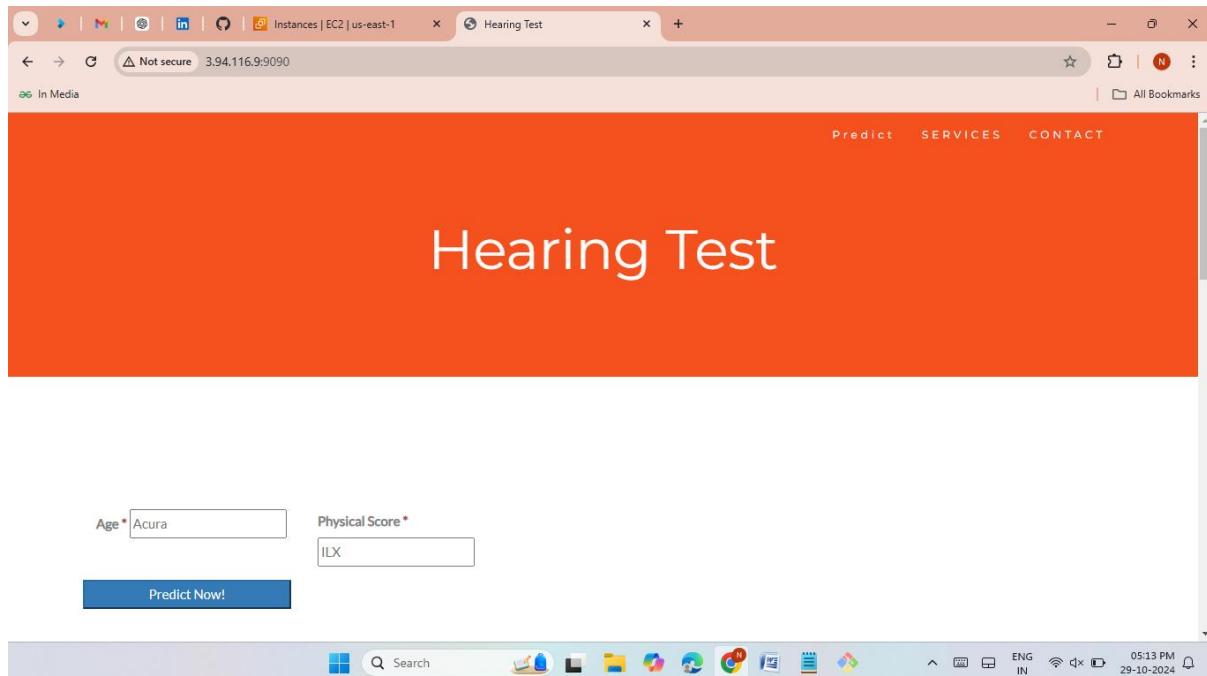
- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the Indian-liver-patients is displays.



- Now clone the hearing python web applications.
- Now install requirements packages with the command.
pip3 install -r requirements.txt
- Now run hearing with the command.
python3 app.py or screen -m -d python3 app.py

```
ec2-user@ip-172-31-18-125:~/hearing
[ec2-user@ip-172-31-18-125 ~]$ ls
hearing
[ec2-user@ip-172-31-18-125 ~]$ cd hearing/
[ec2-user@ip-172-31-18-125 hearing]$ ls
app.py  first_gaitattributes  Logistic_Regression.ipynb  model.pkl  Procfile  requirements.txt  scaler_joblib  templates
[ec2-user@ip-172-31-18-125 hearing]$ screen -m -d python3 app.py
[ec2-user@ip-172-31-18-125 hearing]$ python3 app.py
/home/ec2-user/.local/lib/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle estimator LogisticRegression from version 1.1.2 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning,
/home/ec2-user/.local/lib/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle estimator StandardScaler from version 1.1.2 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning,
* Serving Flask app 'app'
* Debug mode: on
Address already in use
Port 8080 is in use by another program. Either identify and stop that program, or start the server with a different port.
[ec2-user@ip-172-31-18-125 hearing]$ sudo vi app.py
[ec2-user@ip-172-31-18-125 hearing]$ screen -m -d python3 app.py
[ec2-user@ip-172-31-18-125 hearing]$ python3 app.py
/home/ec2-user/.local/lib/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle estimator LogisticRegression from version 1.1.2 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning,
/home/ec2-user/.local/lib/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle estimator StandardScaler from version 1.1.2 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning,
* Serving Flask app 'app'
* Debug mode: on
Address already in use
```

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the hearing is displays.



- Now clone the Fuel-Consumption-Rating python web applications.
- Now install requirements packages with the command.
pip3 install -r requirements.txt
- Now run Fuel-Consumption-Rating with the command.
python3 app.py or screen -m -d python3 app.py

```
ec2-user@ip-172-31-34-99:~/Fuel-Consumption-Rating
[ec2-user@ip-172-31-34-99 ~]$ git clone https://github.com/Yashwanth-najana/Fuel-Consumption-Rating.git
Cloning into 'Fuel-Consumption-Rating'...
remote: Enumerating objects: 10757, done.
remote: Total 10757 (delta 0), reused 0 (delta 0), pack-reused 10757 (from 1)
Receiving objects: 100% (10757/10757), 81.49 MiB | 24.29 MiB/s, done.
Resolving deltas: 100% (808/808), done.
Updating files: 100% (10267/10267), done.
[ec2-user@ip-172-31-34-99 ~]$ ls
Fuel-Consumption-Rating
[ec2-user@ip-172-31-34-99 ~]$ cd Fuel-Consumption-Rating/
[ec2-user@ip-172-31-34-99 Fuel-Consumption-Rating]$ ls
app.py      fuel          gitattributes      Procfile      RFR.pkl
five.joblib Fuel consumption.ipynb  MY2022 Fuel Consumption Ratings.csv requirements.txt templates
[ec2-user@ip-172-31-34-99 Fuel-Consumption-Rating]$
```

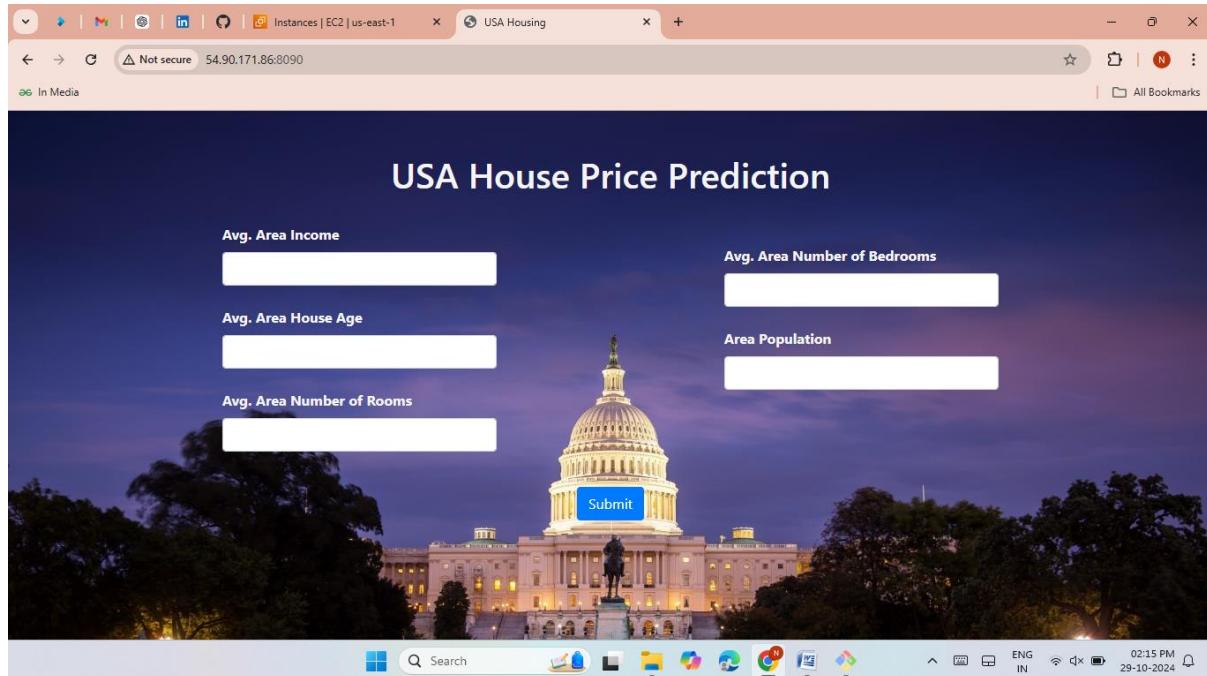
- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the Fuel-Consumption-Rating is displays.

The screenshot shows a web application titled "Fuel Consumption". On the left, there are several dropdown menus and input fields. The dropdowns include "Make" (set to "Acura"), "Model", "Vechile Class" (set to "Compact"), "Transmission" (set to "AM8"), and "Fuel Type". To the right of these dropdowns are input fields for "Engine Size", "Cylinders", "CO2 emission", "CO2 Rating", and "SmokeRate". On the right side of the form, there is a large, detailed image of a fuel gauge with markings from E to F.

- Now clone the USA-Housing python web applications.
- Now install requirements packages with the command.
pip3 install -r requirements.txt
- Now run USA-Housing with the command.
python3 app.py or screen -m -d python3 app.py

```
ec2-user@ip-172-31-34-99:~/USA-Housing
yawsaw@LAPTOP-GM32QJHO MINGW64 ~
$ cd keys
yawsaw@LAPTOP-GM32QJHO MINGW64 ~/keys
$ ssh -i "yash.pem" ec2-user@ec2-54-90-171-86.compute-1.amazonaws.com
The authenticity of host 'ec2-54-90-171-86.compute-1.amazonaws.com (54.90.171.86)' can't be established.
ED25519 key fingerprint is SHA256:GxggEmzbbnDnJwwoOacjih/b96mrGJZNdBkvINF1Xgg.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:395: ec2-54-91-77-142.compute-1.amazonaws.com
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-90-171-86.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Last login: Tue Oct 29 04:38:41 2024 from 171.61.102.221
# 
# ## Amazon Linux 2
## \## AL2 End of Life is 2025-06-30.
## \## 
## \## A newer version of Amazon Linux is available!
## \## / Amazon Linux 2023, GA and supported until 2028-03-15.
## \## https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-172-31-34-99 ~]$ ls
[ec2-user@ip-172-31-34-99 ~]$ git clone https://github.com/Yashwanth-najana/USA-Housing.git
Cloning into 'USA-Housing'...
remote: Enumerating objects: 10754, done.
remote: Total 10754 (delta 0), reused 0 (delta 0), pack-reused 10754 (From 1)
Receiving objects: 100% (10754/10754), 81.02 MiB | 24.30 MiB/s, done.
Resolving deltas: 100% (811/811), done.
Updating files: 100% (10266/10266), done.
[ec2-user@ip-172-31-34-99 ~]$ ls
USA-Housing
[ec2-user@ip-172-31-34-99 ~]$ cd USA-Housing/
[ec2-user@ip-172-31-34-99 USA-Housing]$ ls
app.py gitattributes Price Proffile regressor.pkl requirements.txt templates USA House.ipynb
[ec2-user@ip-172-31-34-99 USA-Housing]$ 
```

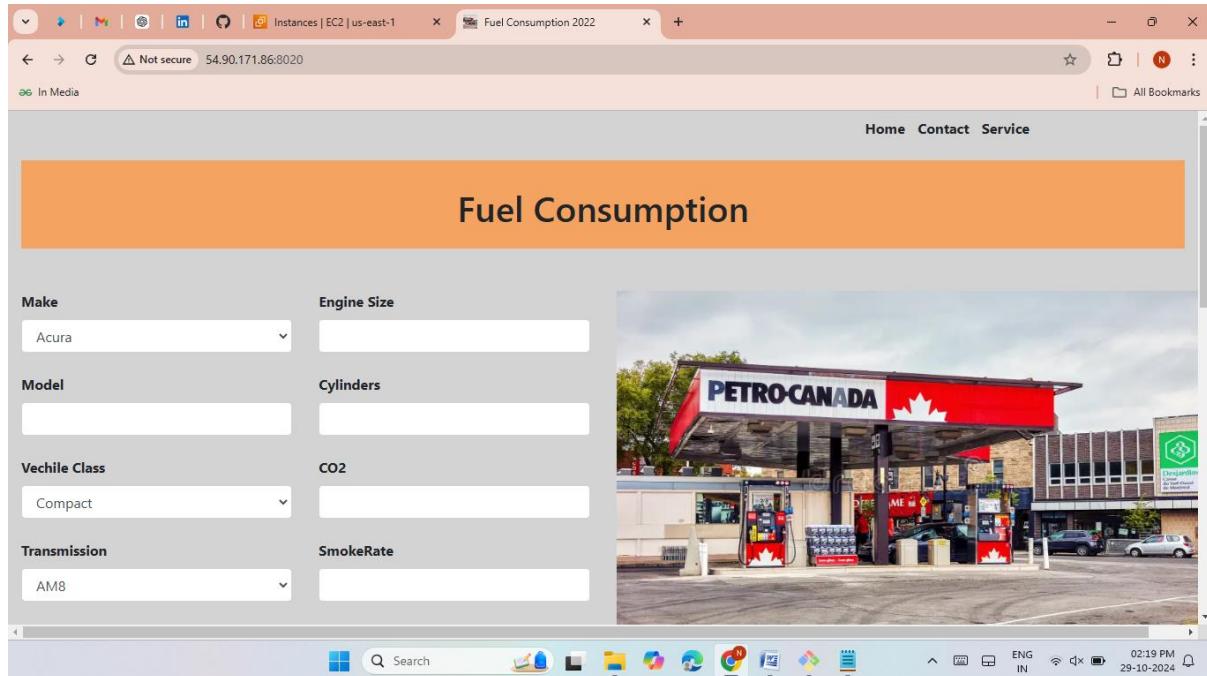
- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the USA-Housing is displays.



- Now clone the MyFuel python web applications.
- Now install requirements packages with the command.
pip3 install -r requirements.txt
- Now run MyFuel with the command.
python3 app.py or screen -m -d python3 app.py

```
ec2-user@ip-172-31-34-99:~/MyFuel
[ec2-user@ip-172-31-34-99 ~]$ git clone https://github.com/Yashwanth-najana/MyFuel.git
Cloning into 'MyFuel'...
remote: Enumerating objects: 10847, done.
remote: Total 10847 (delta 0), reused 0 (delta 0), pack-reused 10847 (from 1)
Receiving objects: 100% (10847/10847), 81.89 MiB | 23.97 MiB/s, done.
Resolving deltas: 100% (814/814), done.
Updating files: 100% (10356/10356), done.
[ec2-user@ip-172-31-34-99 ~]$ ls
MyFuel
[ec2-user@ip-172-31-34-99 ~]$ cd MyFuel/
[ec2-user@ip-172-31-34-99 MyFuel]$ ls
app.py gitattributes MYFuel 2022.ipynb Procfile requirements.txt
Fuel MY2022 Fuel Consumption Ratings.csv OneHotee.joblib regressor.pkl templates
[ec2-user@ip-172-31-34-99 MyFuel]$
```

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the MyFuel is displays.

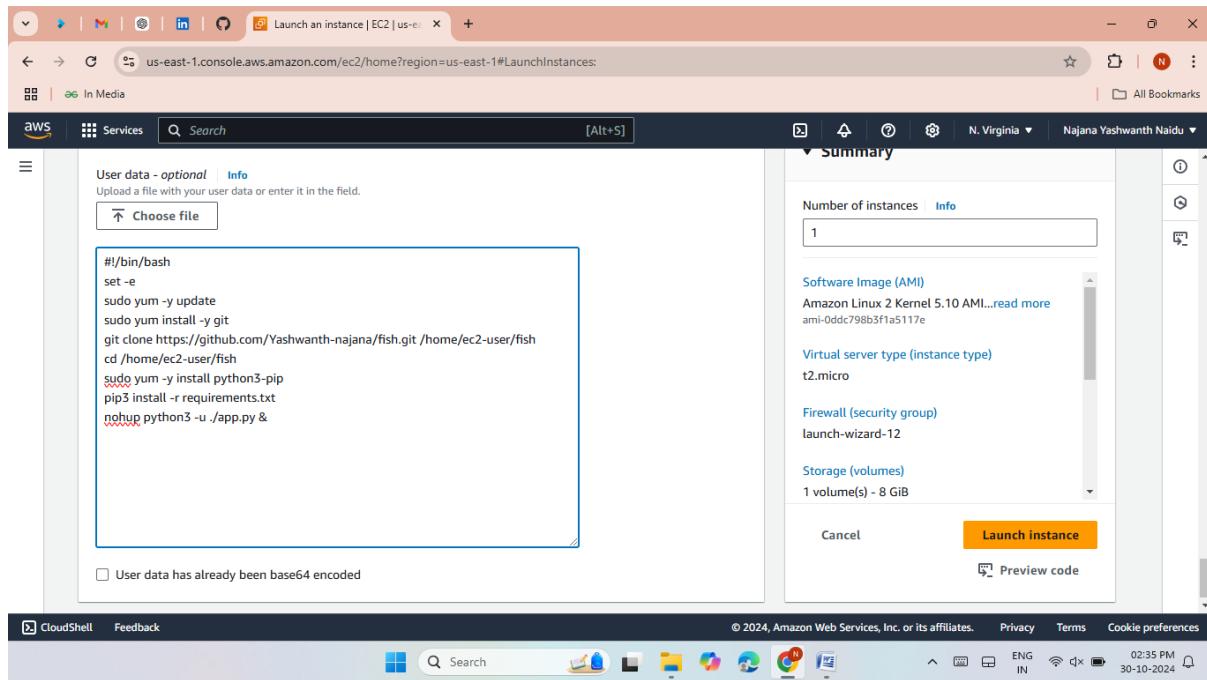


METHOD-2

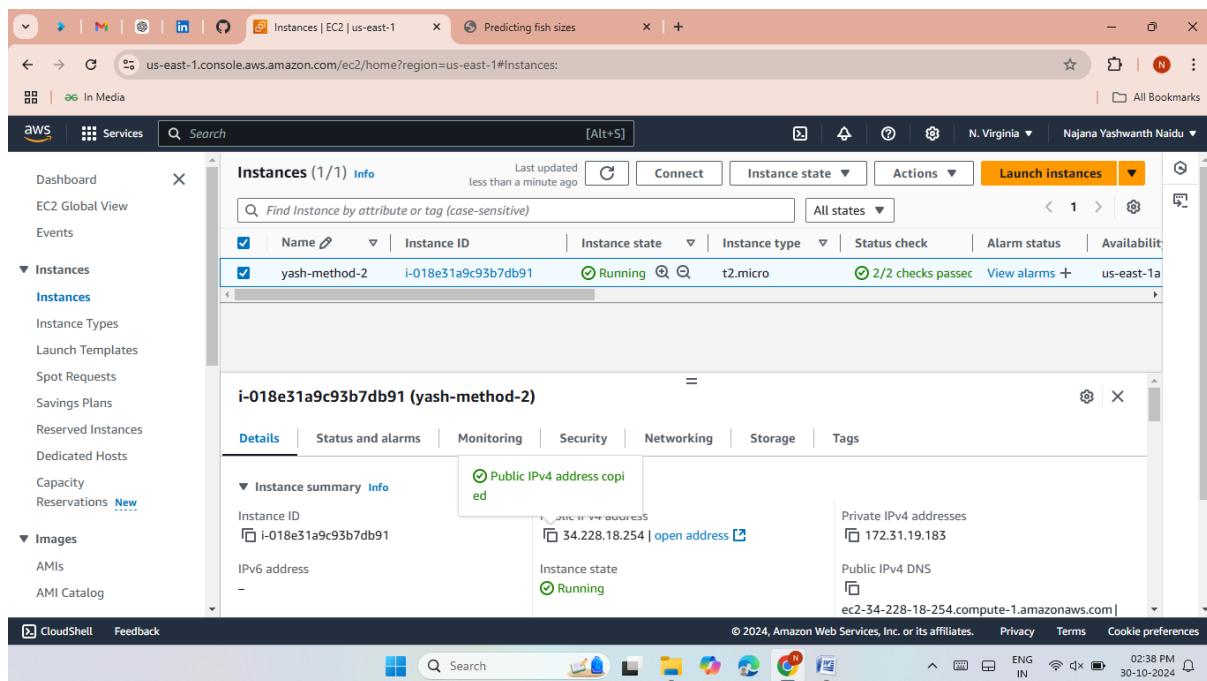
Build and deploy python applications along with EC2 instance(userdata).

- First launch an ec2 instances with the user data script.

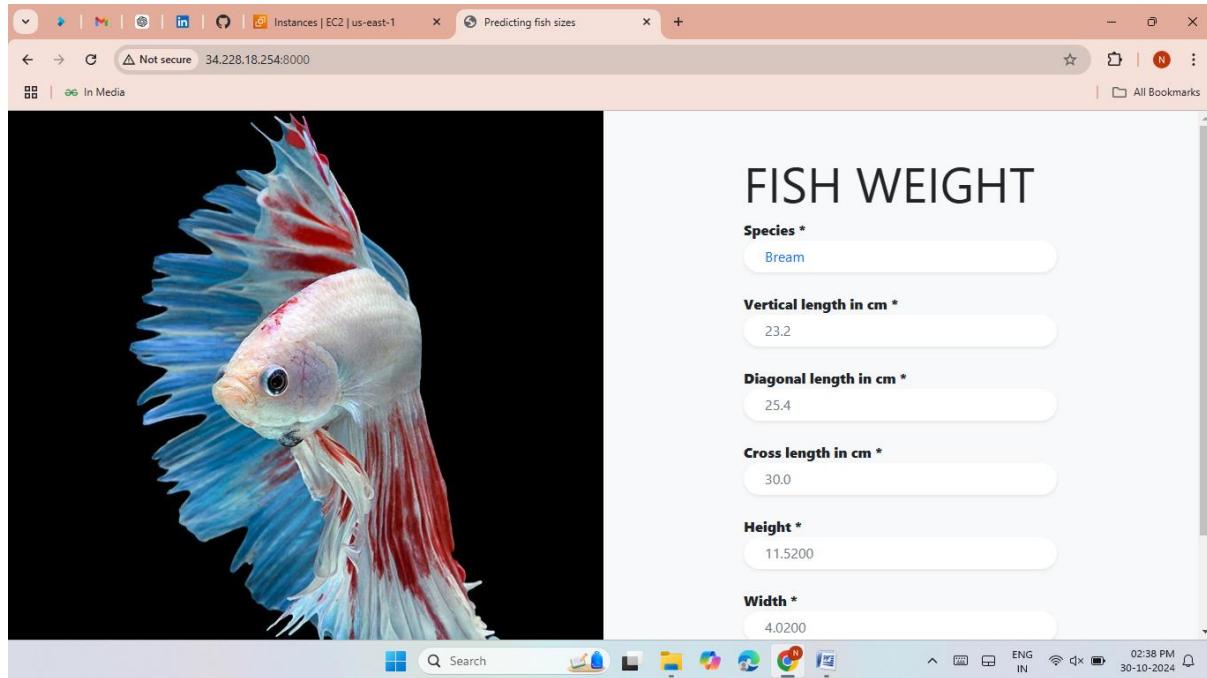
```
#!/bin/bash
set -e
sudo yum -y update
sudo yum install -y git
git clone https://github.com/Yashwanth-najana/fish.git /home/ec2-user/fish
cd /home/ec2-user/fish
sudo yum -y install python3-pip
pip3 install -r requirements.txt
nohup python3 -u ./app.py &
```



➤ Here the instance was launched with the user data.



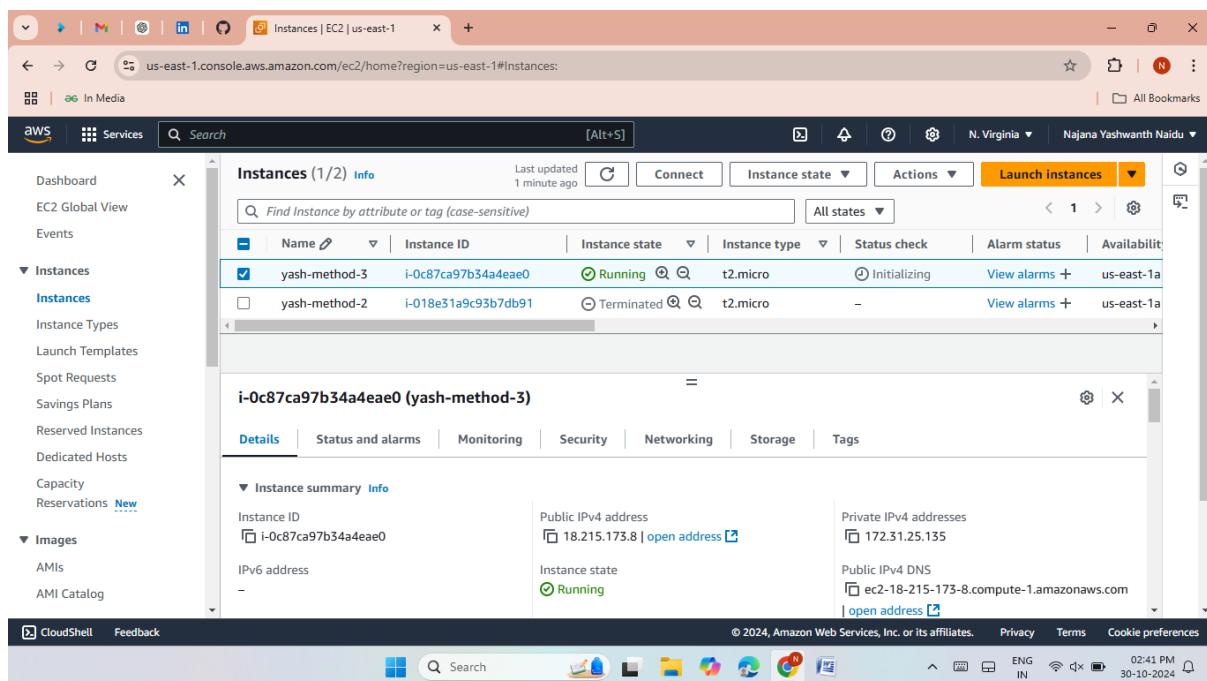
- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the fish is displays.



METHOD-3

Build and deploy python applications in Bash scripting

- First launch an EC2 instance with the required ports.



➤ Now write the bash script for deploy the python application with **sudo vi** mode.

```
#!/bin/bash
sudo yum -y update
sudo yum install -y git
git clone https://github.com/yashwanth-najana/fish.git /home/ec2-user/fish
cd /home/ec2-user/fish
sudo yum -y install python3-pip
pip3 install -r requirements.txt
nohup python3 -u ./app.py &
~
```

-- INSERT --

8,12 All

02:43 PM 30-10-2024 ENG IN

➤ Now run the bash script with the command **./<file name>.sh**

```
ec2-user@ip-172-31-25-135:~/.keys
$ ssh -i 'yash.pem' ec2-user@ec2-18-215-173-8.compute-1.amazonaws.com
The authenticity of host 'ec2-18-215-173-8.compute-1.amazonaws.com (18.215.173.8)' can't be established.
ED25519 key fingerprint is SHA256:WEjGRnAydgHn29GYzPk61Y1PjAM/1FBVqXPKKfekQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: permanently added 'ec2-18-215-173-8.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

Amazon Linux 2
AL2 End of Life is 2025-06-30.
A newer version of Amazon Linux is available!
Amazon Linux 2023, GA and supported until 2028-03-15.
https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-172-31-25-135 ~]$ sudo vi data.sh
[ec2-user@ip-172-31-25-135 ~]$ ls
data.sh
[ec2-user@ip-172-31-25-135 ~]$ sudo chmod +x data.sh
[ec2-user@ip-172-31-25-135 ~]$ ./data.sh
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
No packages marked for update
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
--> Package git.x86_64 0:2.40.1-1.amzn2.0.3 will be installed
--> Processing Dependency: git-core = 2.40.1-1.amzn2.0.3 for package: git-2.40.1-1.amzn2.0.3.x86_64
--> Processing Dependency: git-core-doc = 2.40.1-1.amzn2.0.3 for package: git-2.40.1-1.amzn2.0.3.x86_64
--> Processing Dependency: perl-Git = 2.40.1-1.amzn2.0.3 for package: git-2.40.1-1.amzn2.0.3.x86_64
--> Processing Dependency: perl(Git) for package: git-2.40.1-1.amzn2.0.3.x86_64
--> Processing Dependency: perl(Term::ReadKey) for package: git-2.40.1-1.amzn2.0.3.x86_64
--> Running transaction check
--> Package git-core.x86_64 0:2.40.1-1.amzn2.0.3 will be installed
--> Package git-core-doc.noarch 0:2.40.1-1.amzn2.0.3 will be installed
--> Processing Dependency: perl-Git.noarch 0:2.40.1-1.amzn2.0.3 will be installed
--> Processing Dependency: perl(Error) for package: perl-Git-2.40.1-1.amzn2.0.3.noarch
--> Package perl-TermReadkey.x86_64 0:2.30-20.amzn2.0.2 will be installed
--> Running transaction check
--> Package perl-Error.noarch 1:0.17020-2.amzn2 will be installed
--> Finished Dependency Resolution

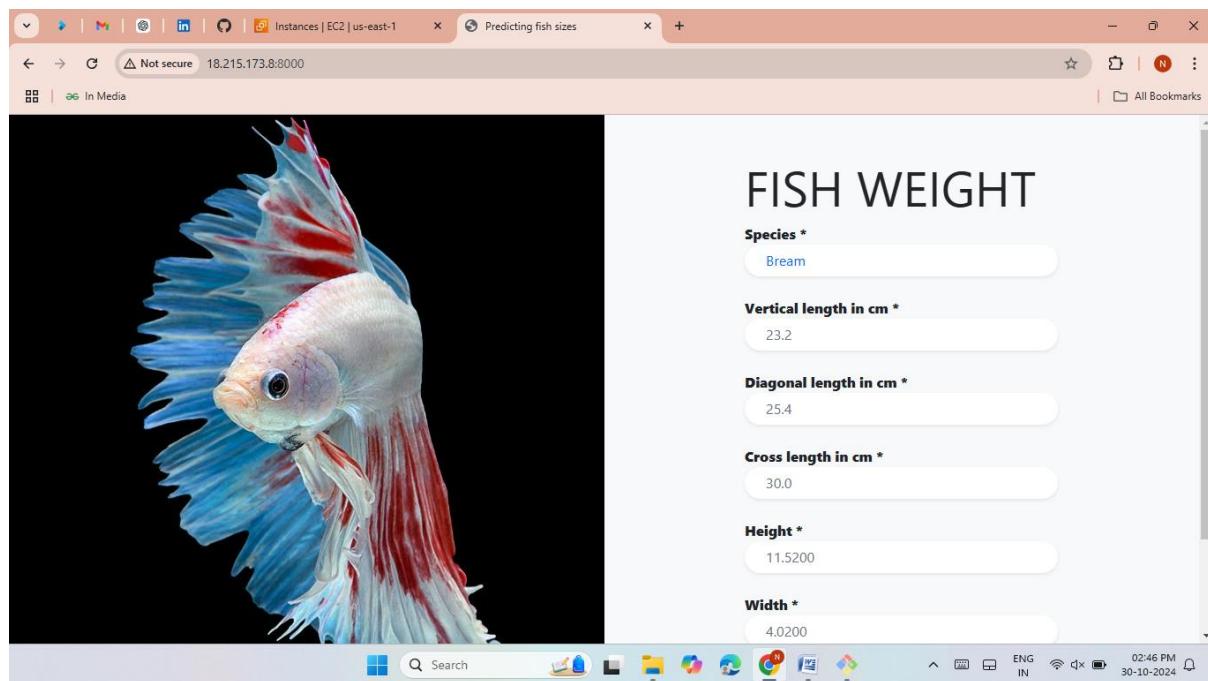
Dependencies Resolved
```

02:44 PM 30-10-2024 ENG IN

```

ec2-user@ip-172-31-25-135:~ 
  Downloading importlib_metadata-6.7.0-py3-none-any.whl (22 kB)
Collecting itsdangerous
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting Jinja2
  Downloading Jinja2-3.1.4-py3-none-any.whl (133 kB)
    |██████████| 133 kB 20.7 MB/s
Collecting joblib
  Downloading joblib-1.3.2-py3-none-any.whl (302 kB)
    |██████████| 302 kB 20.6 MB/s
Collecting MarkupSafe
  Downloading MarkupSafe-2.1.5-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (25 kB)
Collecting numpy
  Downloading numpy-1.21.6-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (15.7 MB)
    |██████████| 15.7 MB 37.5 MB/s
Collecting pandas
  Downloading pandas-1.3.5-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.3 MB)
    |██████████| 11.3 MB 22.4 MB/s
Collecting python-dateutil
  Downloading python_dateutil-2.9.0.post0-py3-none-any.whl (229 kB)
    |██████████| 229 kB 45.2 MB/s
Collecting pytz
  Downloading pytz-2024.2-py2.py3-none-any.whl (508 kB)
    |██████████| 508 kB 38.2 MB/s
Collecting scikit-learn
  Downloading scikit_learn-1.0.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (24.8 MB)
    |██████████| 24.8 MB 23.6 MB/s
Collecting scipy
  Downloading scipy-1.7.3-cp37-cp37m-manylinux_2_17_x86_64.manylinux2010_x86_64.whl (38.1 MB)
    |██████████| 38.1 MB 213 kB/s
Collecting six
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Collecting threadpoolctl
  Downloading threadpoolctl-3.1.0-py3-none-any.whl (14 kB)
Collecting werkzeug
  Downloading Werkzeug-2.2.3-py3-none-any.whl (238 kB)
    |██████████| 238 kB 36.0 MB/s
Collecting zipp
  Downloading zipp-3.15.0-py3-none-any.whl (6.8 kB)
Collecting gunicorn
  Downloading gunicorn-23.0.0-py3-none-any.whl (85 kB)
    |██████████| 85 kB 7.4 MB/s
Collecting typing_extensions<3.6.4; python_version < "3.8"
  Downloading typing_extensions-4.7.1-py3-none-any.whl (33 kB)
Collecting packaging
  Downloading packaging-24.0-py3-none-any.whl (53 kB)
    |██████████| 53 kB 4.4 MB/s
Installing collected packages: typing_extensions, importlib-metadata, click, colorama, MarkupSafe, werkzeug, itsdangerous, Jinja2, Flask, joblib, numpy, six, python-dateutil, pytz, pandas, scipy, threadpoolctl, scikit-learn, packaging, gunicorn
Successfully installed Flask-2.2.2 Jinja2-3.1.4 MarkupSafe-2.1.5 Werkzeug-2.2.3 click-8.1.7 colorama-0.4.6 gunicorn-23.0.0 importlib-metadata-6.7.0 itsdangerous-2.1.2 j
oblib-1.3.2 numpy-1.21.6 packaging-24.0 pandas-1.3.5 python-dateutil-2.9.0.post0 pytz-2024.2 scikit-learn-1.0.2 scipy-1.7.3 six-1.16.0 threadpoolctl-3.1.0 typing-extens
ions-4.7.1 zipp-3.15.0
[ec2-user@ip-172-31-25-135 ~]$ nohup: appending output to 'nohup.out'
[ec2-user@ip-172-31-25-135 ~]$
```

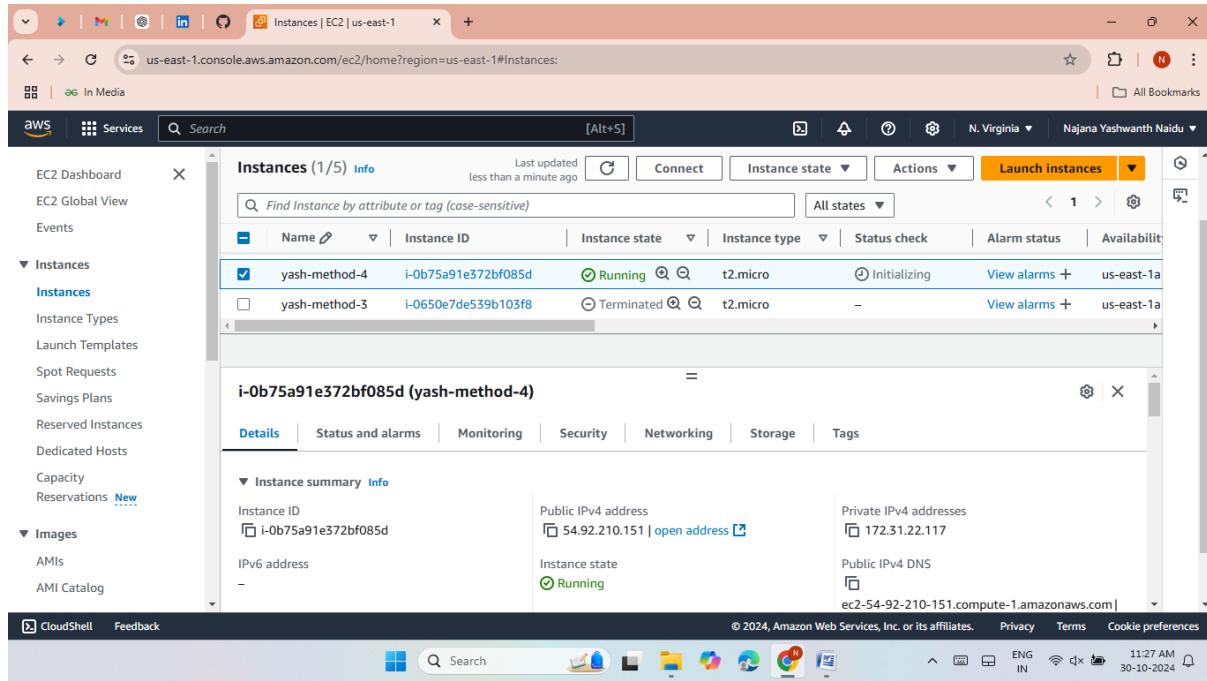
- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the fish is displays.



METHOD-4

Build and deploy python applications with Git, Github and Jenkins (execute shell).

- Launch EC2 instance with Jenkins port number(8080) and given python applications port.



- Now install Jenkins and then start and enable with the commands.

```
sudo yum install -y java-17*
sudo wget -O /etc/yum.repos.d/jenkins.repo \
https://pkg.jenkins.io/redhat/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat/jenkins.io-2023.key
sudo yum upgrade -y
sudo yum install -y jenkins
sudo systemctl start Jenkins
sudo systemctl enable jenkins
```

```

ec2-user@ip-172-31-22-117:~$ 
Downloading packages:
jenkins-2.483-1.1.noarch.rpm
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : jenkins-2.483-1.1.noarch
  Verifying   : jenkins-2.483-1.1.noarch

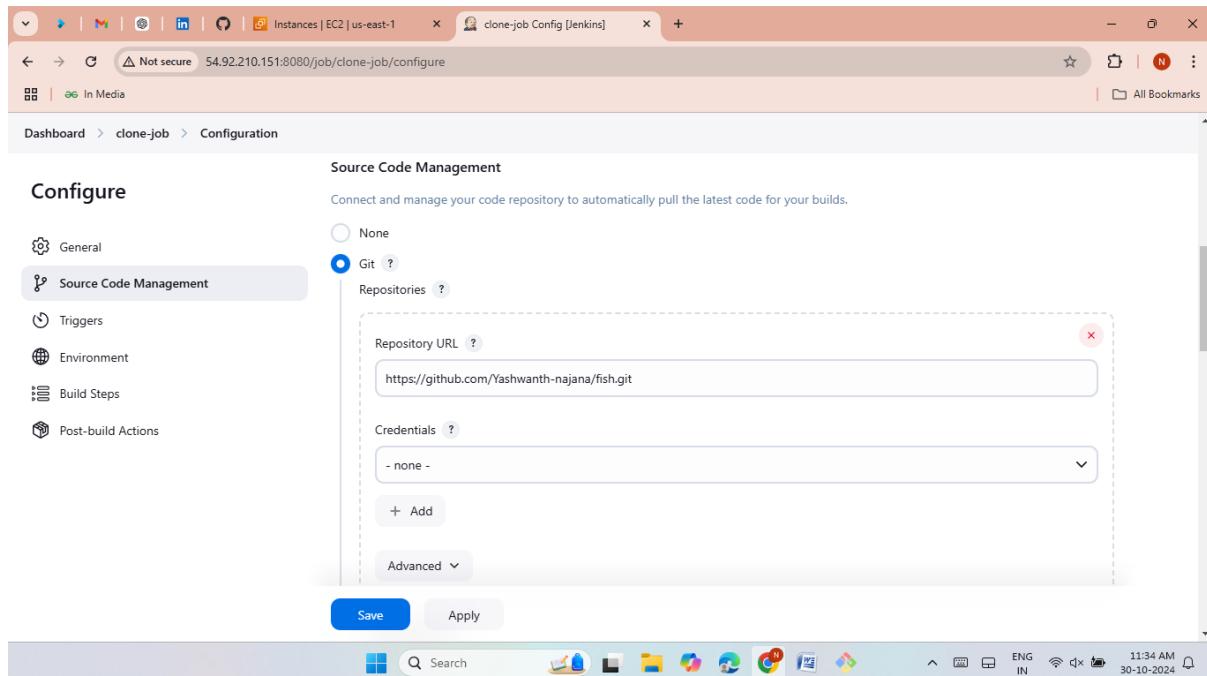
Installed:
  jenkins.noarch 0:2.483-1.1

Complete!
[ec2-user@ip-172-31-22-117 ~]$ sudo systemctl start jenkins
[ec2-user@ip-172-31-22-117 ~]$ sudo systemctl enable jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service to /usr/lib/systemd/system/jenkins.service.
[ec2-user@ip-172-31-22-117 ~]$ sudo systemctl status jenkins
● Jenkins.service - Jenkins Continuous Integration Server
  Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; vendor preset: disabled)
  Active: active (running) since Wed 2024-10-30 06:01:25 UTC; 14s ago
    Main PID: 4003 (Java)
   CGroup: /system.slice/jenkins.service
           └─4003 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080

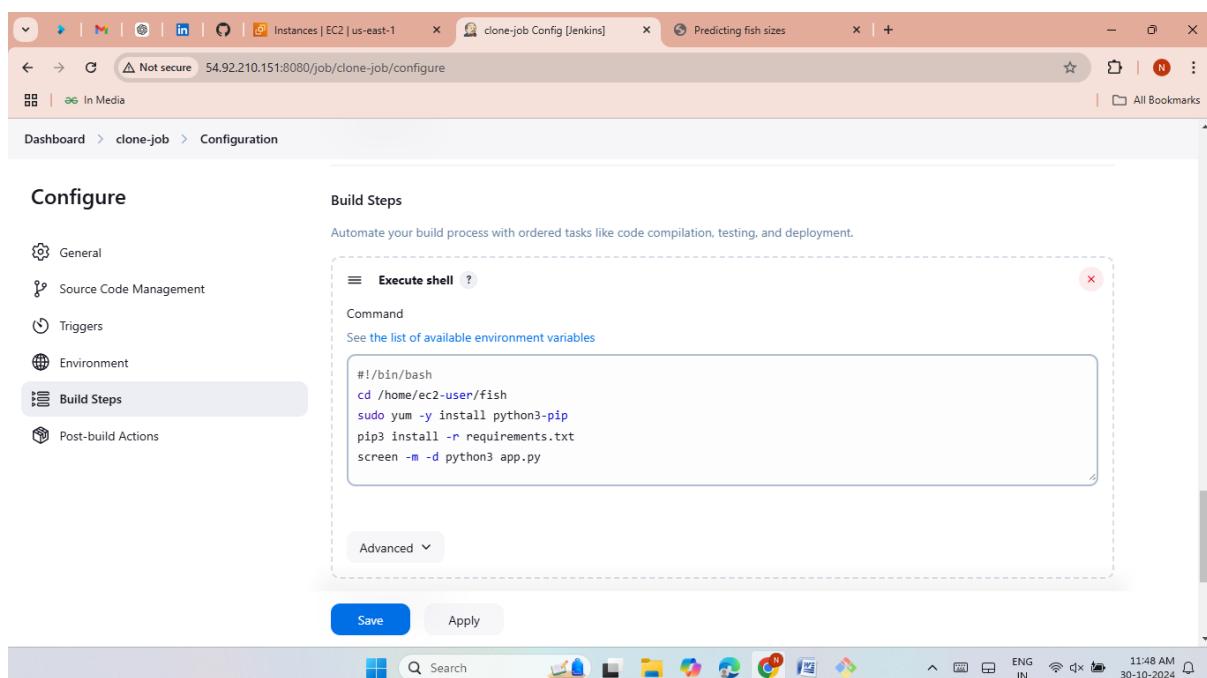
Oct 30 06:01:19 ip-172-31-22-117.ec2.internal jenkins[4003]: ****
Oct 30 06:01:19 ip-172-31-22-117.ec2.internal jenkins[4003]: ****
Oct 30 06:01:19 ip-172-31-22-117.ec2.internal jenkins[4003]: ****
Oct 30 06:01:25 ip-172-31-22-117.ec2.internal jenkins[4003]: 2024-10-30 06:01:25.728+0000 [id=32]      INFO      jenkins.I...ation
Oct 30 06:01:25 ip-172-31-22-117.ec2.internal jenkins[4003]: 2024-10-30 06:01:25.773+0000 [id=24]      INFO      hudson.li...ning
Oct 30 06:01:25 ip-172-31-22-117.ec2.internal systemd[1]: Started Jenkins Continuous Integration Server.
Oct 30 06:01:25 ip-172-31-22-117.ec2.internal jenkins[4003]: 2024-10-30 06:01:25.826+0000 [id=47]      INFO      h.m.Downl...aller
Oct 30 06:01:25 ip-172-31-22-117.ec2.internal jenkins[4003]: 2024-10-30 06:01:25.827+0000 [id=47]      INFO      hudson.ut...pt #1
Oct 30 06:01:33 ip-172-31-22-117.ec2.internal systemd[1]: [/usr/lib/systemd/system/jenkins.service:16] unknown lvalue 'StartLi...Unit'
Oct 30 06:01:33 ip-172-31-22-117.ec2.internal systemd[1]: [/usr/lib/systemd/system/jenkins.service:17] unknown lvalue 'StartLi...Unit'
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-22-117 ~]$ 
```

➤ Here the Jenkins was launched.

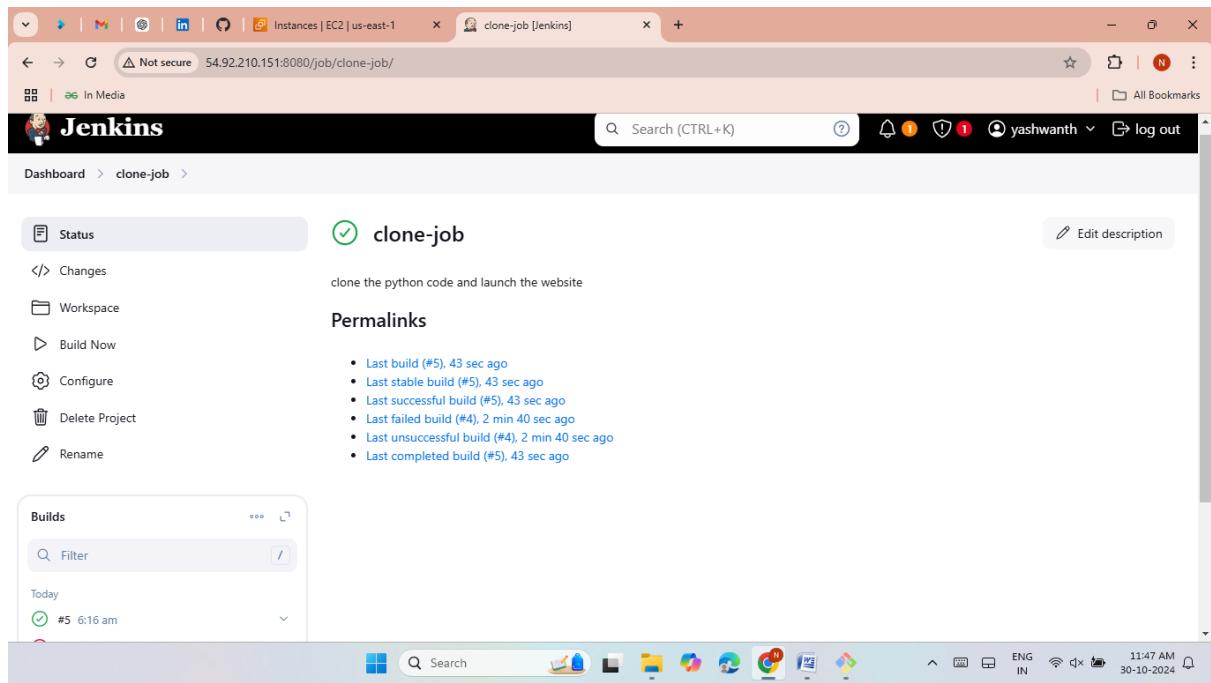
- Create a job for cloned the repo.
- Now Clone the repository URL.



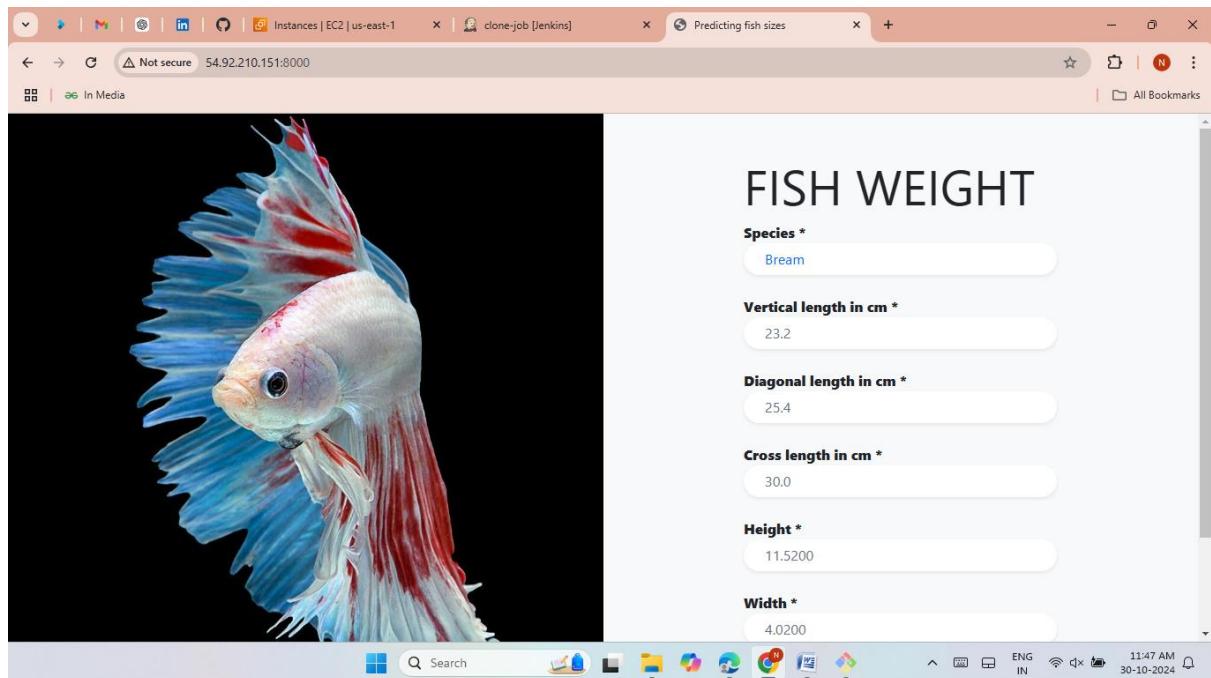
- Now write the script in execute shell.



- Now click on build. Then job is success.



- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the fish python application is displays.

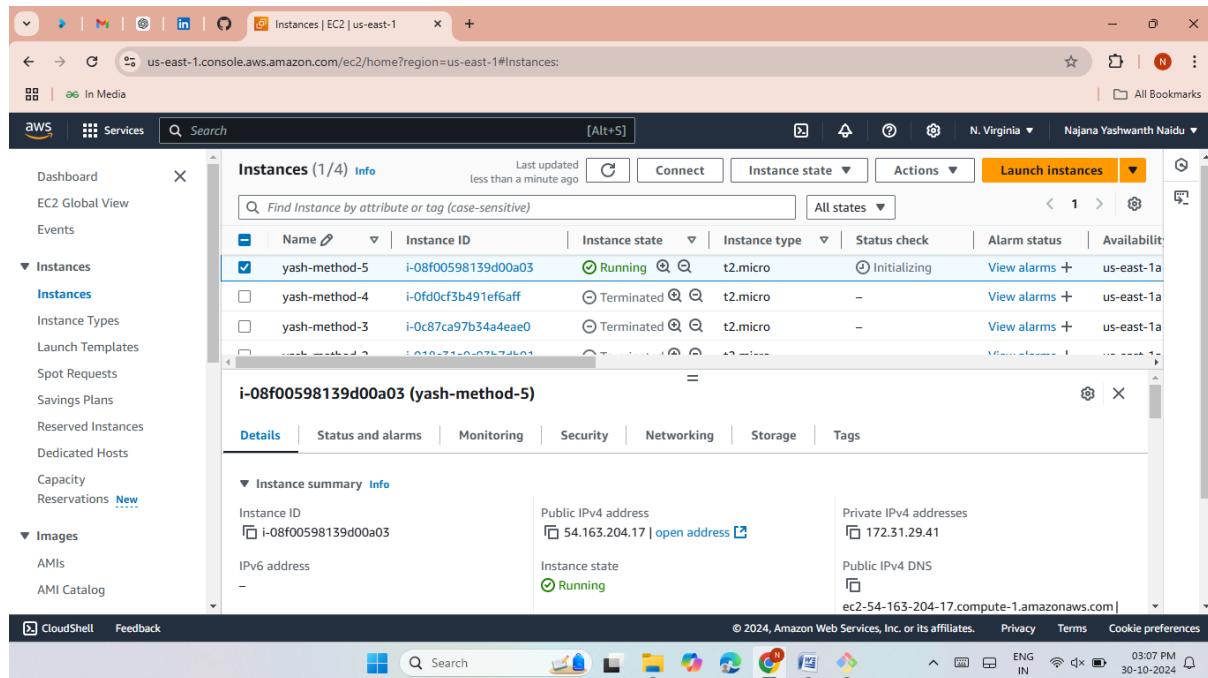


METHOD-5

Build and deploy python applications with the Terraform(data.sh) and push the Terraform

Scripted files in Github.

- First Launch an EC2 instance in our AWS account.



- Now install terraform from Google browser.

```
ec2-user@ip-172-31-29-41:~ - 03:10 PM 30-10-2024
(4/7): perl-Error-0.17020-2.amzn2.noarch.rpm | 32 kB 00:00:00
(5/7): perl-Git-2.40.1-1.amzn2.0.3.noarch.rpm | 42 kB 00:00:00
(6/7): perl-TermReadKey-2.30-20.amzn2.0.2.x86_64.rpm | 31 kB 00:00:00
warning: /var/cache/yum/x86_64/2/hashicorp/packages/terraform-1.9.8-1.x86_64.rpm: Header V4 RSA/SHA256 Signature, key ID a621e701: NOKEY
Public key for terraform-1.9.8-1.x86_64.rpm is not installed
(7/7): terraform-1.9.8-1.x86_64.rpm | 27 MB 00:00:00
Total                                         66 MB/s | 40 MB 00:00:00
Retrieving key from https://rpm.releases.hashicorp.com/gpg
Importing GPG key 0xA621E701:
Userid   : HashiCorp Security (HashiCorp Package Signing) <security+packaging@hashicorp.com>
Fingerprint: 798a ec65 4e5c 1542 8c8e 4aa6 fcfc a621 e701
From     : https://rpm.releases.hashicorp.com/gpg
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : git-core-2.40.1-1.amzn2.0.3.x86_64          1/7
  Installing : git-core-doc-2.40.1-1.amzn2.0.3.noarch      2/7
  Installing : l:perl-Error-0.17020-2.amzn2.noarch        3/7
  Installing : perl-TermReadkey-2.30-20.amzn2.0.2.x86_64  4/7
  Installing : perl-Git-2.40.1-1.amzn2.0.3.noarch        5/7
  Installing : git-2.40.1-1.amzn2.0.3.x86_64            6/7
  Verifying   : perl-TermReadkey-2.30-20.amzn2.0.2.x86_64  1/7
  Verifying   : terraform-1.9.8-1.x86_64                  2/7
  Verifying   : git-2.40.1-1.amzn2.0.3.x86_64            3/7
  Verifying   : l:perl-Error-0.17020-2.amzn2.noarch        4/7
  Verifying   : git-core-2.40.1-1.amzn2.0.3.x86_64        5/7
  Verifying   : git-core-doc-2.40.1-1.amzn2.0.3.noarch      6/7
  Verifying   : perl-Git-2.40.1-1.amzn2.0.3.noarch        7/7

Installed:
  terraform.x86_64 0:1.9.8-1

Dependency Installed:
  git.x86_64 0:2.40.1-1.amzn2.0.3           git-core.x86_64 0:2.40.1-1.amzn2.0.3
  perl-Error.noarch 0:1.17020-2.amzn2          perl-Git.noarch 0:2.40.1-1.amzn2.0.3           git-core-doc.noarch 0:2.40.1-1.amzn2.0.3
                                                               perl-TermReadkey.x86_64 0:2.30-20.amzn2.0.2

Complete!
[ec2-user@ip-172-31-29-41 ~]$ |
```

- Write a terraform script for launch an EC2 instance.
- Provide Access key and Secret key.
- Write script for EC2 and Security Groups, VPC, Subnets, Internet Gateway, Route Table.

```
ec2-user@ip-172-31-29-41:~$ provider "aws" {
  region = "us-east-1"
  access_key = ""
  secret_key = ""
}

#creating VPC
resource "aws_vpc" "yashvpc" {
  cidr_block = "10.0.0.0/16"
  instance_tenancy = "default"
  tags = {
    Name = "yashvpc"
  }
}
#subnet_creation

resource "aws_subnet" "web-subnet1" {
  vpc_id      = aws_vpc.yashvpc.id
  cidr_block  = "10.0.1.0/24"
  map_public_ip_on_launch = true
  availability_zone = "us-east-1a"
  tags = {
    name = "web-subnet1"
  }
}
resource "aws_subnet" "web-subnet2" {
  vpc_id      = aws_vpc.yashvpc.id
  cidr_block  = "10.0.2.0/24"
  map_public_ip_on_launch = true
  availability_zone = "us-east-1b"
  tags = {
    name = "web-subnet2"
  }
}
resource "aws_subnet" "application-subnet1" {
  vpc_id      = aws_vpc.yashvpc.id
  cidr_block  = "10.0.3.0/24"
  map_public_ip_on_launch = true
  availability_zone = "us-east-1a"
  tags = {
    name = "application-subnet1"
  }
}
resource "aws_subnet" "application-subnet2" {
  vpc_id      = aws_vpc.yashvpc.id
  cidr_block  = "10.0.4.0/24"
  map_public_ip_on_launch = true
}
-- INSERT --
```

```
ec2-user@ip-172-31-29-41:~$#creating internet gateway
resource "aws_internet_gateway" "yashigw" {
  vpc_id = aws_vpc.yashvpc.id
}
#Creating route table
resource "aws_route_table" "public-route-table" {
  vpc_id = aws_vpc.yashvpc.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.yashigw.id
  }
  tags = {
    Name = "public-route-table"
  }
}

# Associating Route table
resource "aws_route_table_association" "rt1" {
  subnet_id   = aws_subnet.web-subnet1.id
  route_table_id = aws_route_table.public-route-table.id
}

# Associating Route table
resource "aws_route_table_association" "rt2" {
  subnet_id   = aws_subnet.web-subnet2.id
  route_table_id = aws_route_table.public-route-table.id
}

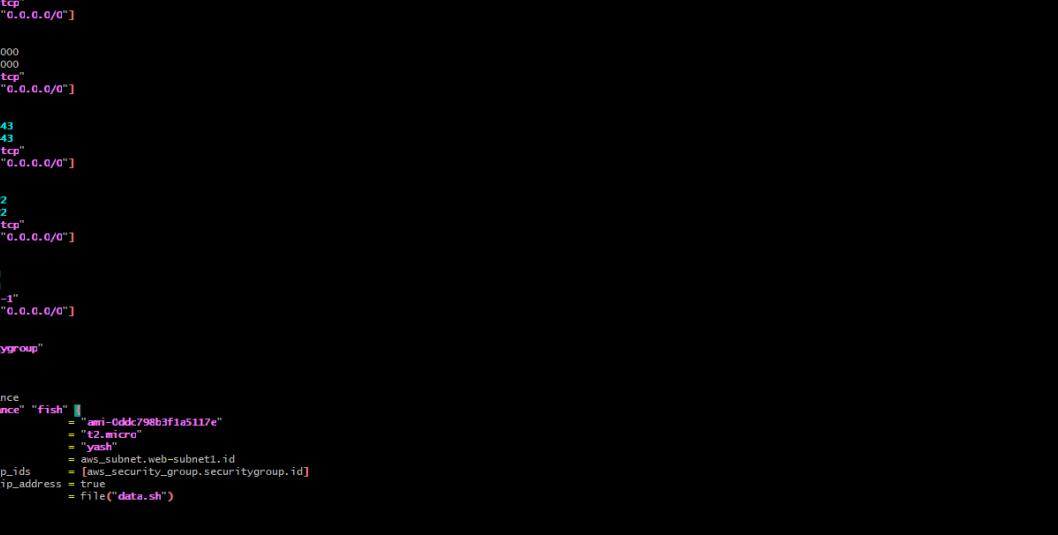
#creating security group
resource "aws_security_group" "securitygroup" {
  vpc_id = aws_vpc.yashvpc.id
  #Inbound Rules
  ingress {
    from_port  = 80
    to_port    = 80
    protocol   = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    from_port  = 8000
    to_port    = 8000
    protocol   = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    from_port  = 443
    to_port    = 443
    protocol   = "tcp"
  }
}
-- INSERT --
```

```
ec2-user@ip-172-31-29-41:~
```

```
ingress {
    from_port = 80
    to_port = 80
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
}
ingress {
    from_port = 8000
    to_port = 8000
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
}
ingress {
    from_port = 443
    to_port = 443
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
}
ingress {
    from_port = 22
    to_port = 22
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
}
egress {
    from_port = 0
    to_port = 0
    protocol = "-1"
    cidr_blocks = ["0.0.0.0/0"]
}
tags = {
    Name = "securitygroup"
}

#Creating EC2 instance
resource "aws_instance" "fish" {
    ami = "ami-0ddc798b3f1a5117e"
    instance_type = "t2.micro"
    key_name = "yash"
    subnet_id = aws_subnet.web-subnet1.id
    vpc_security_group_ids = [aws_security_group.securitygroup.id]
    associate_public_ip_address = true
    user_data = file("data.sh")
}
tags = {
    Name = "yash"
}
```

-- INSERT --



The screenshot shows a Linux desktop environment with a terminal window open. The terminal displays AWS CloudFormation configuration code for creating an EC2 instance and defining security groups. Below the terminal, the desktop interface includes a file browser, system status icons (battery, signal, etc.), and a dock with various application icons like a file manager, terminal, and browser.

- Now write the script for data file.
 - Clone the fish python application form github.

➤ Now Enter the following commands

```
terraform init  
terraform fmt  
terraform validate  
terraform plan  
terraform apply
```

```
[ec2-user@ip-172-31-29-41:~]  
[ec2-user@ip-172-31-29-41 ~]$ ls  
data.sh python.tf  
[ec2-user@ip-172-31-29-41 ~]$ terraform init  
Initializing the backend...  
Initializing provider plugins...  
- Finding latest version of hashicorp/aws...  
- Installing hashicorp/aws v5.73.0...  
- Installed hashicorp/aws v5.73.0 (signed by HashiCorp)  
Terraform has created a lock file .terraform.lock.hcl to record the provider  
selections it made above. Include this file in your version control repository  
so that Terraform can guarantee to make the same selections by default when  
you run "terraform init" in the future.  
Terraform has been successfully initialized!  
You may now begin working with Terraform. Try running "terraform plan" to see  
any changes that are required for your infrastructure. All Terraform commands  
should now work.  
If you ever set or change modules or backend configuration for Terraform,  
rerun this command to reinitialize your working directory. If you forget, other  
commands will detect it and remind you to do so if necessary.  
[ec2-user@ip-172-31-29-41 ~]$ terraform fmt  
python.tf  
Error: Failed to write python.tf  
  
[ec2-user@ip-172-31-29-41 ~]$ sudo vi python.tf  
[ec2-user@ip-172-31-29-41 ~]$ sudo terraform fmt  
python.tf  
[ec2-user@ip-172-31-29-41 ~]$ sudo terraform validate  
Success! The configuration is valid.  
[ec2-user@ip-172-31-29-41 ~]$
```

```
[ec2-user@ip-172-31-29-41:~]  
+ "name" = "web-subnet2"  
} = (known after apply)  
+ vpc_id  
}  
  
# aws_vpc.yashvpc will be created  
+ resource "aws_vpc" "yashvpc" {  
+ arn = (known after apply)  
+ cidr_block = "10.0.0.0/16"  
+ default_network_acl_id = (known after apply)  
+ default_route_table_id = (known after apply)  
+ default_security_group_id = (known after apply)  
+ dhcp_options_id = (known after apply)  
+ enable_dns_hostnames = (known after apply)  
+ enable_dns_support = true  
+ enable_network_address_usage_metrics = (known after apply)  
+ id = (known after apply)  
+ instance_tenancy = "default"  
+ ipv6_association_id = (known after apply)  
+ ipv6_cidr_block = (known after apply)  
+ ipv6_cidr_block_network_border_group = (known after apply)  
+ main_route_table_id = (known after apply)  
+ owner_id = (known after apply)  
+ tags = {  
+ "Name" = "yashvpc"  
}  
+ tags_all = {  
+ "Name" = "yashvpc"  
}  
}  
  
Plan: 11 to add, 0 to change, 0 to destroy.  
  
Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run  
"terraform apply" now.  
[ec2-user@ip-172-31-29-41 ~]$ |
```

```

ec2-user@ip-172-31-29-41:~ 
Plan: 11 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_vpc.yashvpc: Creating...
aws_vpc.yashvpc: Creation complete after 1s [id=vpc-097d2dfd6e49f1f79]
aws_subnet.application-subnet1: Creating...
aws_subnet.web-subnet1: Creating...
aws_subnet.application-subnet2: Creating...
aws_internet_gateway.yashigw: Creating...
aws_security_group.securitygroup: Creating...
aws_subnet.web-subnet2: Creating...
aws_internet_gateway.yashigw: Creation complete after 1s [id=igw-0916415a867e598a5]
aws_route_table.public-route-table: Creating...
aws_route_table.public-route-table: Creation complete after 1s [id=rtb-0b82e6295b2de7217]
aws_security_group.securitygroup: Creation complete after 3s [id=sg-0323e37d8219082d7]
aws_subnet.application-subnet1: Still creating... [10s elapsed]
aws_subnet.web-subnet1: Still creating... [10s elapsed]
aws_subnet.application-subnet2: Still creating... [10s elapsed]
aws_subnet.application-subnet1: Creation complete after 11s [id=subnet-0206943421fe6268f]
aws_subnet.application-subnet2: Creation complete after 11s [id=subnet-04be354581c7d1c68]
aws_subnet.web-subnet2: Creation complete after 11s [id=subnet-03d8398a3ead4f37]
aws_route_table_association.rt2: Creating...
aws_subnet.web-subnet1: Creation complete after 11s [id=subnet-0c37d18ed4851e99a]
aws_instance.fish: Creating...
aws_route_table_association.rt1: Creating...
aws_route_table_association.rt2: Creation complete after 1s [id=rtbassoc-094c76f439f2c4b3c]
aws_route_table_association.rt1: Creation complete after 1s [id=rtbassoc-0b5f1521ff10f0865]
aws_instance.fish: Still creating... [10s elapsed]
aws_instance.fish: Creation complete after 13s [id=i-00f57d1de5032ea90]

Apply complete! Resources: 11 added, 0 changed, 0 destroyed.
[ec2-user@ip-172-31-29-41 ~]$ 

```

➤ Here the EC2 instance was launched.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
yash-metrooc-4	i-00f57d1de5032ea90	Terminated	t2.micro	-	View alarms	us-east-1a
yash	i-00f57d1de5032ea90	Running	t2.micro	Initializing	View alarms	us-east-1a
yash-method-3	i-0c87ca97b34a4eae0	Terminated	t2.micro	-	View alarms	us-east-1a
yash-method-2	i-018e31a9c93b7db91	Terminated	t2.micro	-	View alarms	us-east-1a

- Here the VPC, Subnet, Internet Gateway, Route Table, Security Groups also created.

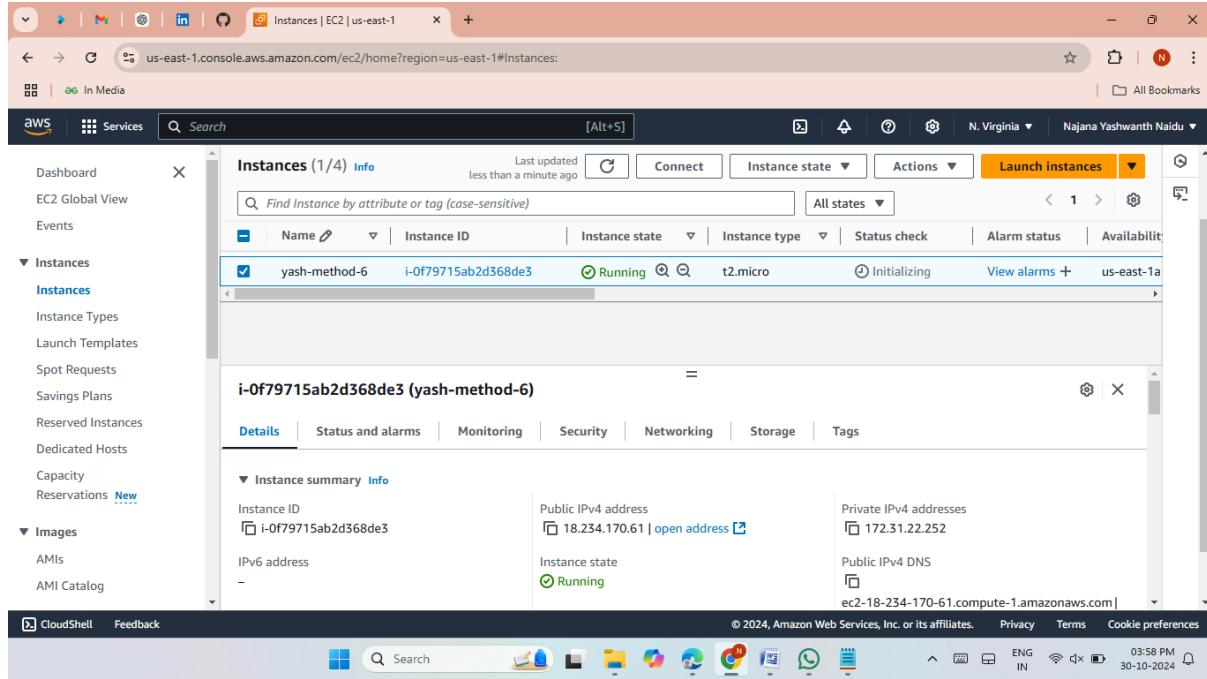
Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR
-	vpc-05f2f9c35f844e371	Available	172.31.0.0/16	-
yashvpc	vpc-097d2dfd6e49f1f79	Available	10.0.0.0/16	-

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the fish python application is displays.

METHOD-6

Build and deploy python applications with Git, github, Jenkins and Terraform

- First launch the EC2 instance with the Jenkins port number(8080) and given required ports.



- Now Install git and Jenkins.
- Now start and enable the Jenkins.

```
ec2-user@ip-172-31-22-252:~$ Total download size: 91 M
Installed size: 92 M
Downloading packages:
jenkins-2.483-1.1.noarch.rpm
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : jenkins-2.483-1.1.noarch
  Verifying : jenkins-2.483-1.1.noarch
Installed:
  jenkins.noarch 0:2.483-1.1
Complete!
[ec2-user@ip-172-31-22-252 ~]$ sudo systemctl start jenkins
[ec2-user@ip-172-31-22-252 ~]$ sudo systemctl enable jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service to /usr/lib/systemd/system/jenkins.service.
[ec2-user@ip-172-31-22-252 ~]$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
  Loaded: Loaded (/usr/lib/systemd/system/jenkins.service; enabled; vendor preset: disabled)
  Active: active (running) since Wed 2024-10-30 10:31:48 UTC; 18s ago
    Main PID: 4074 (java)
   CGroup: /system.slice/jenkins.service
           └─4074 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080

Oct 30 10:31:41 ip-172-31-22-252.ec2.internal jenkins[4074]: ****
Oct 30 10:31:41 ip-172-31-22-252.ec2.internal jenkins[4074]: ****
Oct 30 10:31:41 ip-172-31-22-252.ec2.internal jenkins[4074]: ****
Oct 30 10:31:48 ip-172-31-22-252.ec2.internal jenkins[4074]: 2024-10-30 10:31:48.431+0000 [id=31]      INFO      jenkins.I...ation
Oct 30 10:31:48 ip-172-31-22-252.ec2.internal jenkins[4074]: 2024-10-30 10:31:48.460+0000 [id=24]      INFO      hudson.li...ning
Oct 30 10:31:48 ip-172-31-22-252.ec2.internal systemd[1]: Started Jenkins Continuous Integration Server.
Oct 30 10:31:48 ip-172-31-22-252.ec2.internal jenkins[4074]: 2024-10-30 10:31:48.550+0000 [id=47]      INFO      h.m.Down...aller
Oct 30 10:31:48 ip-172-31-22-252.ec2.internal jenkins[4074]: 2024-10-30 10:31:48.550+0000 [id=47]      INFO      hudson.ut...pt #1
Oct 30 10:31:57 ip-172-31-22-252.ec2.internal systemd[1]: [/usr/lib/systemd/system/jenkins.service:16] unknown lvalue 'StartLi...Unit'
Oct 30 10:31:57 ip-172-31-22-252.ec2.internal systemd[1]: [/usr/lib/systemd/system/jenkins.service:17] unknown lvalue 'StartLi...Unit'
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-22-252 ~]$
```

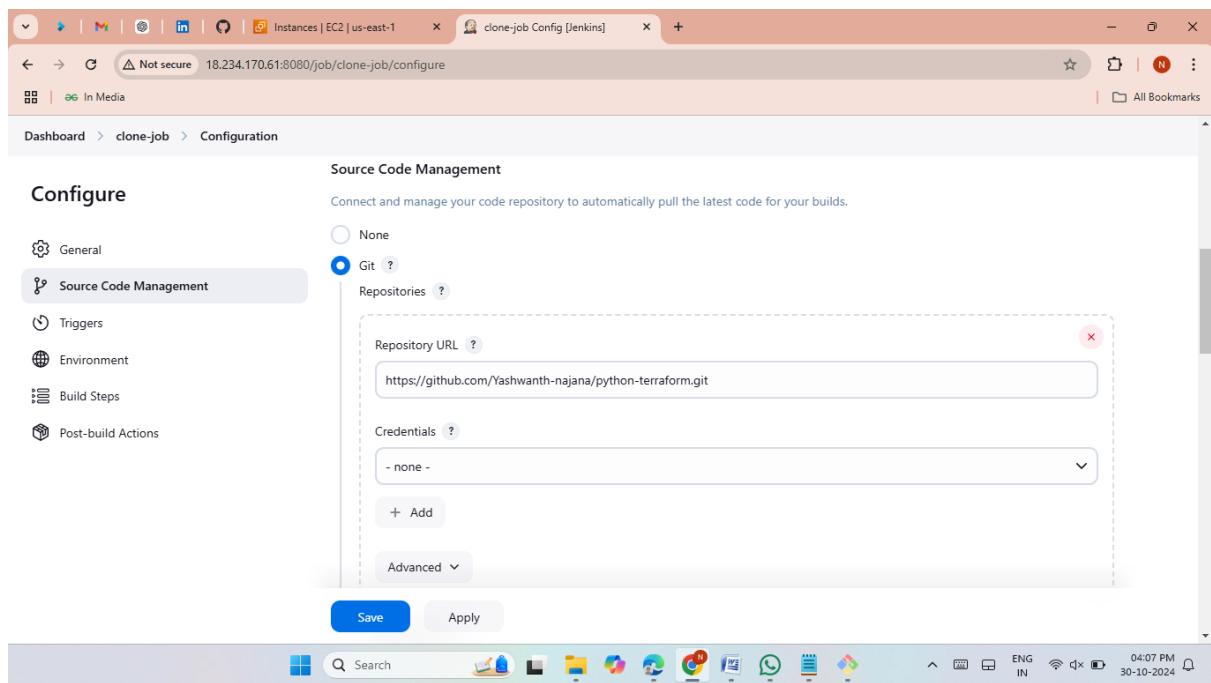
➤ Here the Jenkins was launched successfully.

The screenshot shows the Jenkins dashboard at <http://18.234.170.61:8080>. The main header says "Dashboard [Jenkins]". The left sidebar includes links for "New Item", "Build History", "Manage Jenkins", and "My Views". The central area features a "Welcome to Jenkins!" message and a "Start building your software project" section. It also displays "Build Queue" (empty) and "Build Executor Status" (0/2). Below these are sections for "Set up a distributed build", "Set up an agent", "Configure a cloud", and "Learn more about distributed builds". The bottom status bar shows system information like "ENG IN", "04:04 PM", and the date "30-10-2024".

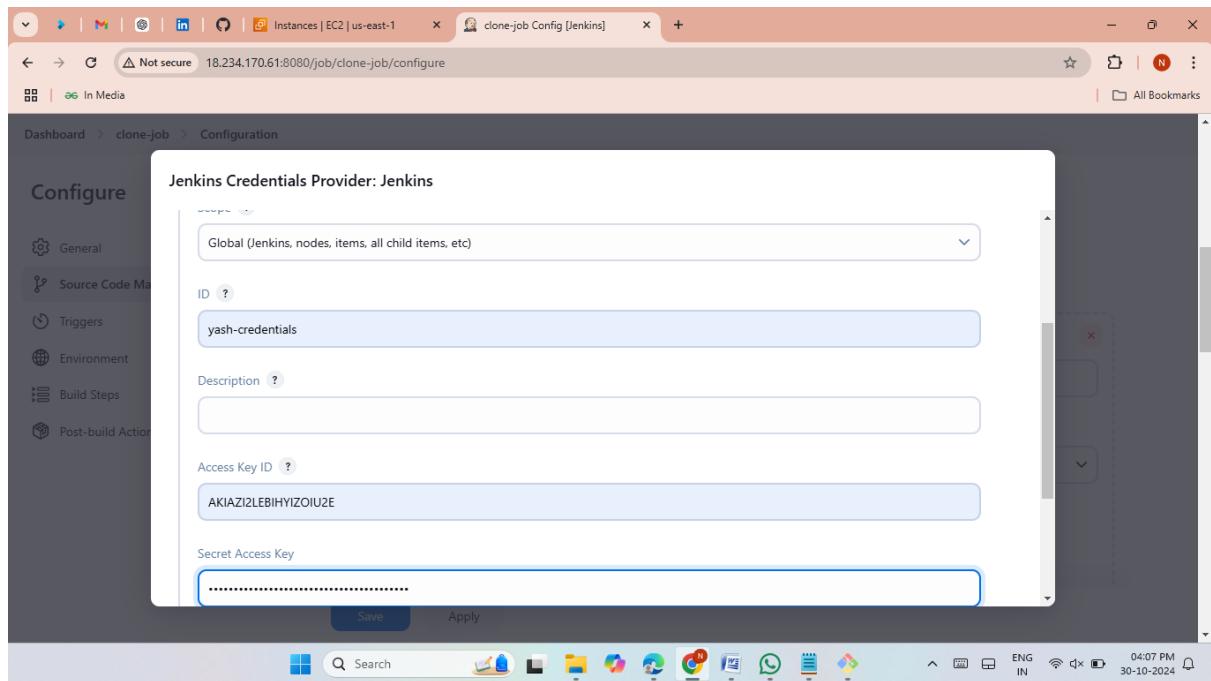
➤ Now install the AWS credentials plugin.

The screenshot shows the Jenkins plugin manager at <http://18.234.170.61:8080/manage/pluginManager/available>. The left sidebar has tabs for "Updates", "Available plugins" (selected), "Installed plugins", "Advanced settings", and "Download progress". A search bar at the top right contains "aws". The main table lists available plugins, with "AWS Credentials" checked for installation. Other listed plugins include "Amazon Web Services SDK :: Minimal" and "Amazon Web Services SDK :: EC2". The bottom status bar shows "04:05 PM" and the date "30-10-2024".

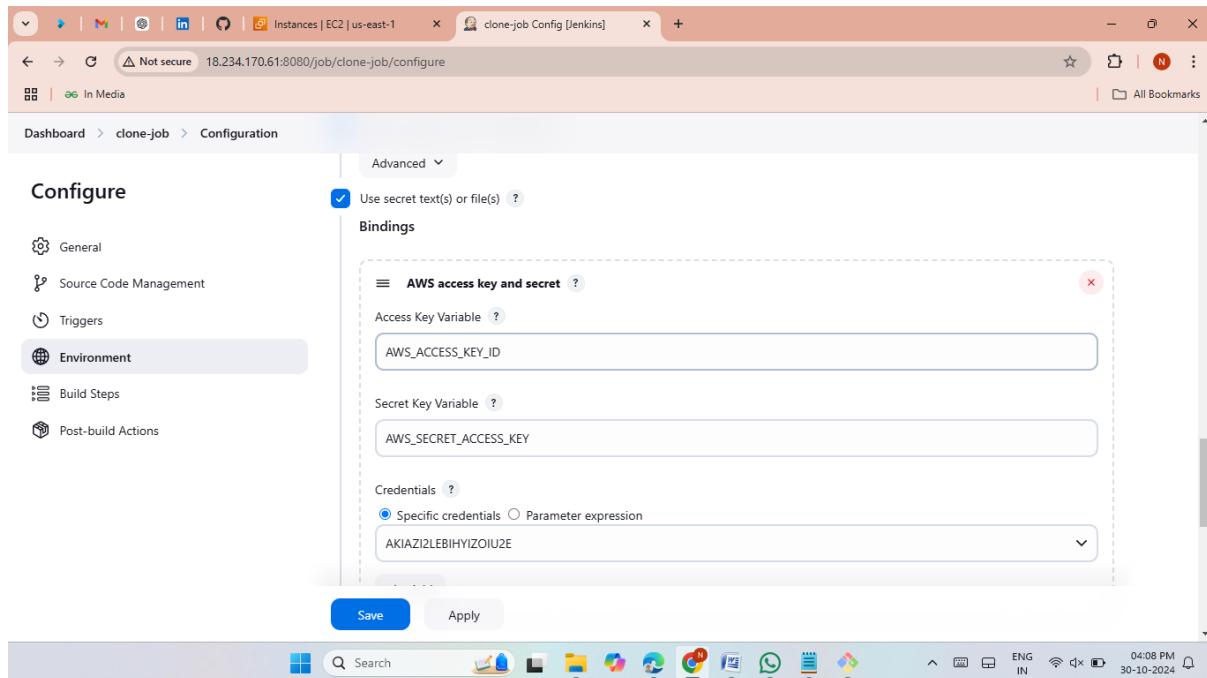
- Now create a job and clone the terraform script from github.



- Then create the credentials.
- Now add the AWS access key and secret key.
- Now apply the changes.



- Now add the AWS access key and secret key.
- Now apply the changes.



- Now install the terraform in execute shell.
- Now add the terraform commands.

```
#!/bin/bash
```

```
sudo yum install -y yum-utils shadow-utils
```

```
sudo yum-config-manager --add-repo
```

```
https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
```

```
sudo yum -y install terraform
```

```
sudo chmod + data.sh
```

```
terraform init
```

```
terraform fmt
```

```
terraform validate
```

```
terraform plan
```

```
terraform apply --auto-approve
```

The screenshot shows the Jenkins job configuration page for a job named 'clone-job'. On the left, there's a sidebar with options like General, Source Code Management, Triggers, Environment, Build Steps (which is selected), and Post-build Actions. The main area is titled 'Build Steps' with the sub-section 'Execute shell'. It contains a command block with the following script:

```
#!/bin/bash
sudo yum install -y yum-utils shadow-utils
sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
sudo yum -y install terraform
sudo chmod +x data.sh
terraform init
terraform fmt
terraform validate
terraform apply --auto-approve
```

At the bottom of the 'Execute shell' section are 'Save' and 'Apply' buttons.

- Now give permissions for Jenkins with the commands
sudo visudo
jenkins ALL=(ALL) NOPASSWD: ALL
- Now restart the Jenkins with the command (**sudo systemctl restart Jenkins**).
- Now click on build the job was success.

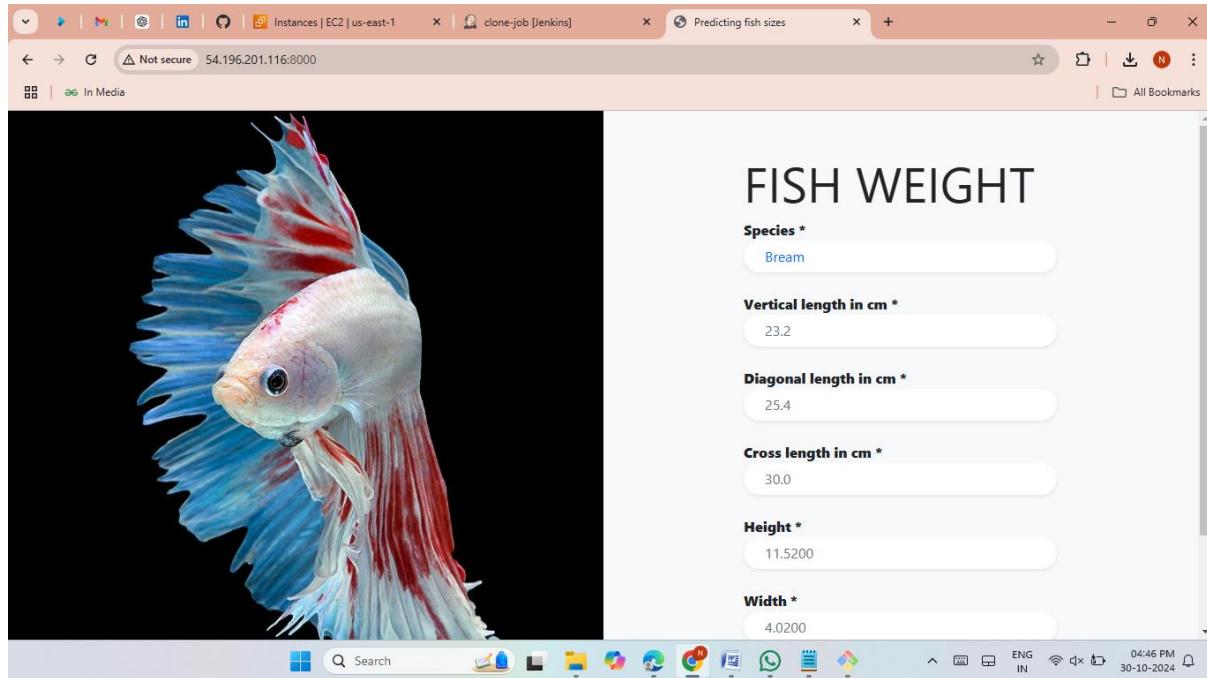
The screenshot shows the Jenkins dashboard for the 'clone-job' project. The top navigation bar includes links for Instances | EC2 | us-east-1, clone-job Config [Jenkins], and clone-job [Jenkins]. The main content area displays the project status as 'Status' (green checkmark) and 'clone-job'. Below this, it says 'clone terraform script and launch python application'. A 'Permalinks' section lists several build links. On the left, there's a sidebar with options like Status, Changes, Workspace, Build Now, Configure, Delete Project, and Rename. At the bottom, there's a 'Builds' section showing a single build entry: '#17 11:13 am'. The system tray at the bottom right shows the date and time as 30-10-2024 04:45 PM.

- Here the EC2 instances was automatically launched with help of terraform script.
- Also create the VPC, Subnet, Internet Gateway, Route Table, Security Groups with help of terraform script.

The screenshot shows the AWS EC2 Instances page. The left sidebar includes options like Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity, and Reservations. The main content area displays a table of instances with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability zone. Two instances are listed: 'yash' (i-0829858eed30de40) is Running and t2.micro, while 'yash-method-6' (i-0f79715ab2d368de3) is also Running and t2.micro. Below the table, a detailed view for instance 'i-0829858eed30de40 (yash)' is shown, including details like Public IPv4 address (54.196.201.116), Private IPv4 addresses (10.0.1.52), and Instance state (Running).

The screenshot shows the AWS VPCs page. The left sidebar includes options like VPC dashboard (selected), EC2 Global View, Filter by VPC, Virtual private cloud (Your VPCs selected), Subnets, Route tables, Internet gateways, Egress-only internet gateways, Carrier gateways, DHCP option sets, Elastic IPs, Managed prefix lists, and Endpoints. The main content area displays a table of VPCs with columns for Name, VPC ID, State, IPv4 CIDR, and IPv6 CIDR. One VPC is listed: 'yashvpc' (vpc-0169505bb8fb9d8a9) is Available with IPv4 CIDR 10.0.0.0/16. Below the table, a detailed view for VPC 'vpc-0169505bb8fb9d8a9 / yashvpc' is shown, including details like VPC ID (vpc-0169505bb8fb9d8a9), State (Available), and DNS resolution (Enabled).

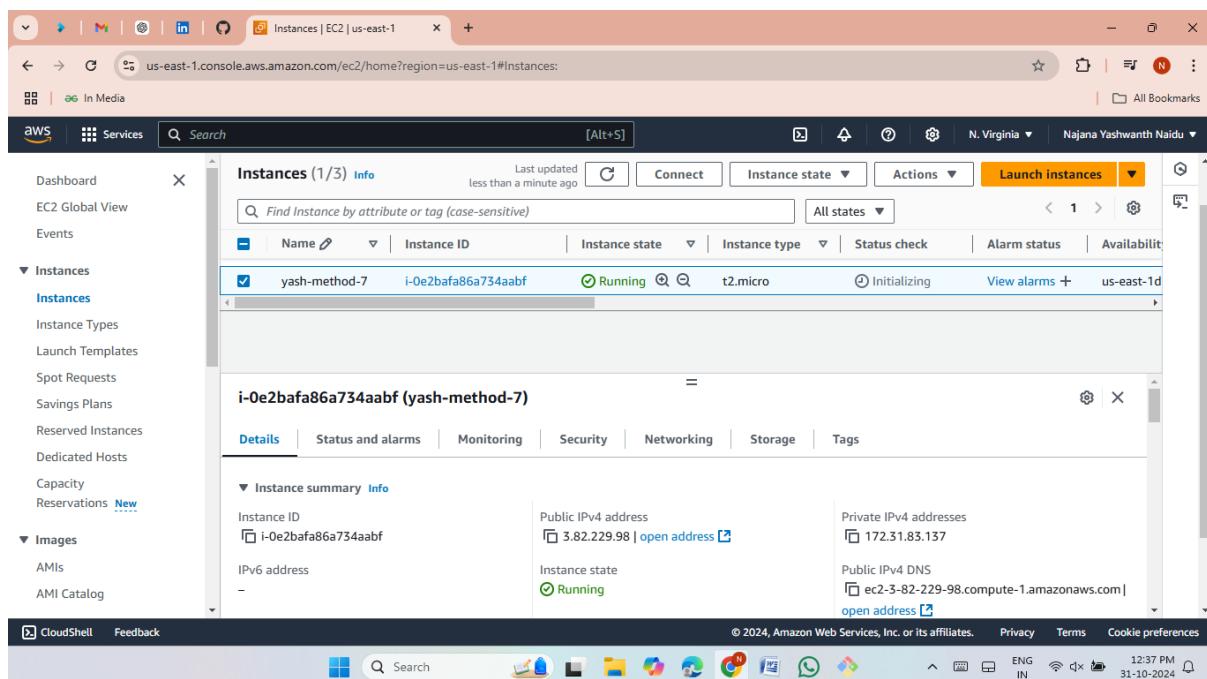
- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the fish python application is displays.



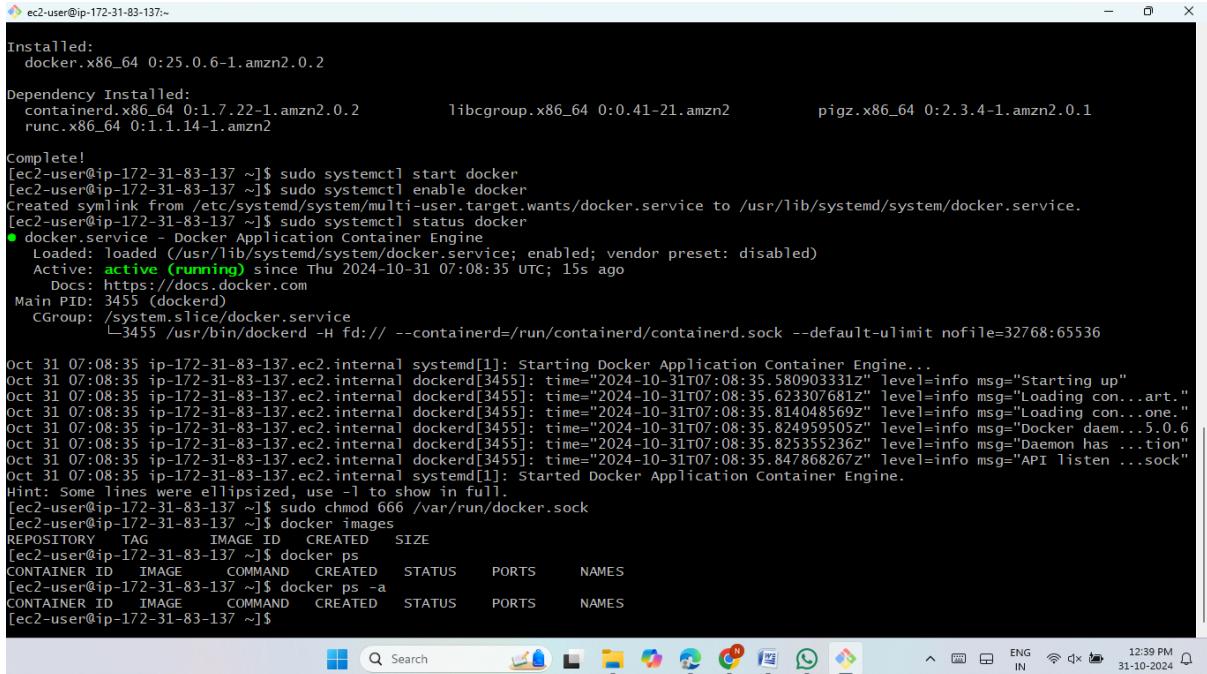
METHOD-7

Build and deploy python applications with Docker (Write docker file and create image from docker file and run containers) and push created docker image in docker hub.

- First launch an EC2 instance.



- Now install docker.
- Now start and enable the docker.
- Give permissions to the docker with the command(**sudo chmod 666 var/run/docker.sock**).
- Now install git.



```

ec2-user@ip-172-31-83-137:~ 
Installed:
  docker.x86_64 0:25.0.6-1.amzn2.0.2

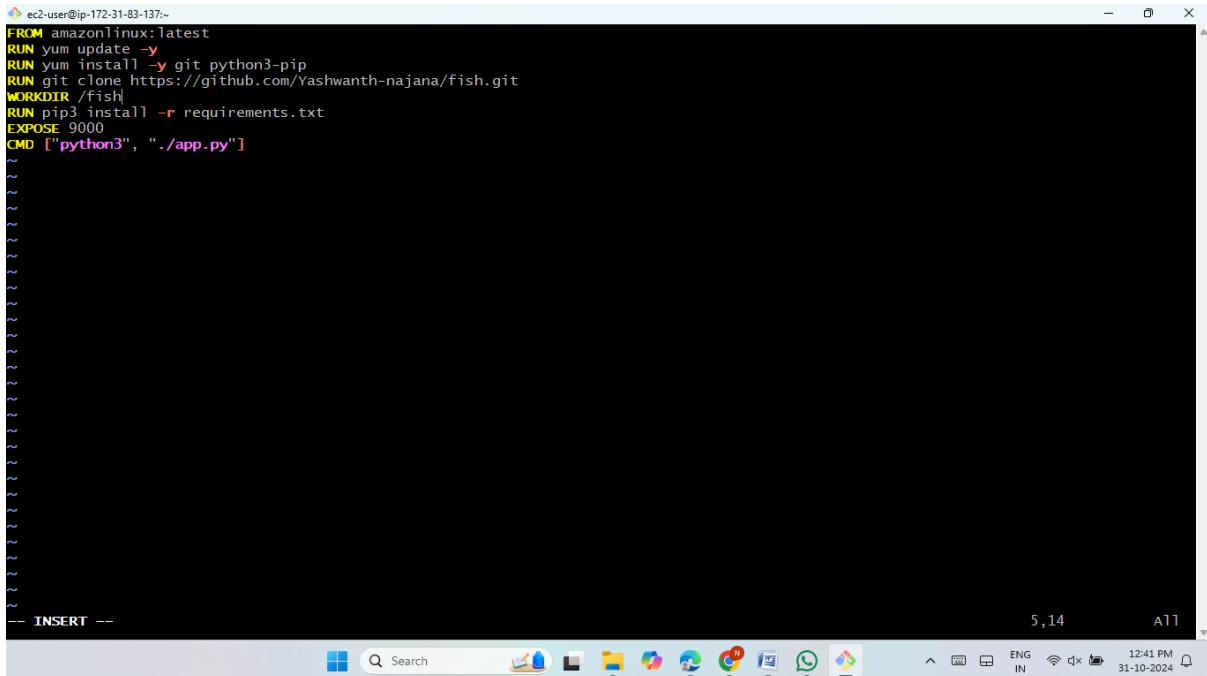
Dependency Installed:
  containerd.x86_64 0:1.7.22-1.amzn2.0.2           libcgroup.x86_64 0:0.41-21.amzn2           pigz.x86_64 0:2.3.4-1.amzn2.0.1

Complete!
[ec2-user@ip-172-31-83-137 ~]$ sudo systemctl start docker
[ec2-user@ip-172-31-83-137 ~]$ sudo systemctl enable docker
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-83-137 ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
  Active: active (running) since Thu 2024-10-31 07:08:35 UTC; 15s ago
    Docs: https://docs.docker.com
  Main PID: 3455 (dockerd)
  CGroup: /system.slice/docker.service
          └─3455 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Oct 31 07:08:35 ip-172-31-83-137.ec2.internal systemd[1]: Starting Docker Application Container Engine...
Oct 31 07:08:35 ip-172-31-83-137.ec2.internal dockerd[3455]: time="2024-10-31T07:08:35.580903331Z" level=info msg="Starting up"
Oct 31 07:08:35 ip-172-31-83-137.ec2.internal dockerd[3455]: time="2024-10-31T07:08:35.623307681Z" level=info msg="Loading con...art."
Oct 31 07:08:35 ip-172-31-83-137.ec2.internal dockerd[3455]: time="2024-10-31T07:08:35.814048569Z" level=info msg="Loading con...one."
Oct 31 07:08:35 ip-172-31-83-137.ec2.internal dockerd[3455]: time="2024-10-31T07:08:35.824959505Z" level=info msg="Docker daem...5.0.6"
Oct 31 07:08:35 ip-172-31-83-137.ec2.internal dockerd[3455]: time="2024-10-31T07:08:35.825355236Z" level=info msg="Daemon has ...tion"
Oct 31 07:08:35 ip-172-31-83-137.ec2.internal dockerd[3455]: time="2024-10-31T07:08:35.847868267Z" level=info msg="API listen ...sock"
Oct 31 07:08:35 ip-172-31-83-137.ec2.internal systemd[1]: Started Docker Application Container Engine.

Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-83-137 ~]$ sudo chmod 666 /var/run/docker.sock
[ec2-user@ip-172-31-83-137 ~]$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
[ec2-user@ip-172-31-83-137 ~]$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
[ec2-user@ip-172-31-83-137 ~]$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
[ec2-user@ip-172-31-83-137 ~]$ 
```

- Now create a docker file and clone the fish python web application.



```

ec2-user@ip-172-31-83-137:~ 
FROM amazonlinux:latest
RUN yum update -y
RUN yum install -y git python3-pip
RUN git clone https://github.com/Yashwanth-najana/fish.git
WORKDIR /fish
RUN pip3 install -r requirements.txt
EXPOSE 9000
CMD ["python3", "./app.py"]

-- INSERT -- 
```

- Now run the following command.

docker build -t <name> .

- Here the docker image was created with help of docker file.

```

ec2-user@ip-172-31-83-137:~$ ls
dockerfile
[ec2-user@ip-172-31-83-137 ~]$ docker build -t fish .
[+] Building 67.9s (10/10) FINISHED
--> [internal] load build definition from dockerfile
--> => transferring dockerfile: 266B
--> [internal] load metadata for docker.io/library/amazonlinux:latest
--> [internal] load .dockerignore
--> => transferring context: 2B
--> [1/6] FROM docker.io/library/amazonlinux:latest@sha256:5bf4cf420ef7e50835911993c6a2ddb0e8f5101c0ef89ca20e9d02a03c8c3a8c
--> => resolve docker.io/library/amazonlinux:latest@sha256:5bf4cf420ef7e50835911993c6a2ddb0e8f5101c0ef89ca20e9d02a03c8c3a8c
--> => sha256:7f7880a52d85d9050dd9997856b2cc315aab1e93e26821536485f5b21dc836c7 1.02kB / 1.02kB
--> => sha256:44dceb23ffa1cef3ee40071b017862c541f285fd263747b54acaafbebc 575B / 575B
--> => sha256:42ce9aa0f68a7f3dc752dad87f21431a084694a3818ff00f932236a9010d564 52.34MB / 52.34MB
--> => sha256:5bf4cf420ef7e50835911993c6a2ddb0e8f5101c0ef89ca20e9d02a03c8c3a8c 2.38kB / 2.38kB
--> => extracting sha256:42ce9aa0f68a7f3dc752dad87f21431a084694a3818ff00f932236a9010d564
--> [2/6] RUN yum update -y
--> [3/6] RUN yum install -y git python3-pip
--> [4/6] RUN git clone https://github.com/yashwanth-najana/fish.git
--> [5/6] WORKDIR /fish
--> [6/6] RUN pip3 install -r requirements.txt
--> => exporting to image
--> => exporting layers
--> => writing image sha256:6215702604d163c7a16cd5e6613ae89ccb1341778c14a7715f94306153dbc22c
--> => naming to docker.io/library/fish
[ec2-user@ip-172-31-83-137 ~]$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
fish latest 6215702604d1 26 seconds ago 1.22GB
[ec2-user@ip-172-31-83-137 ~]$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
[ec2-user@ip-172-31-83-137 ~]$ |

```

- Now run the command.

Docker run -dt --name <name> -p <port>:8000 <image name>

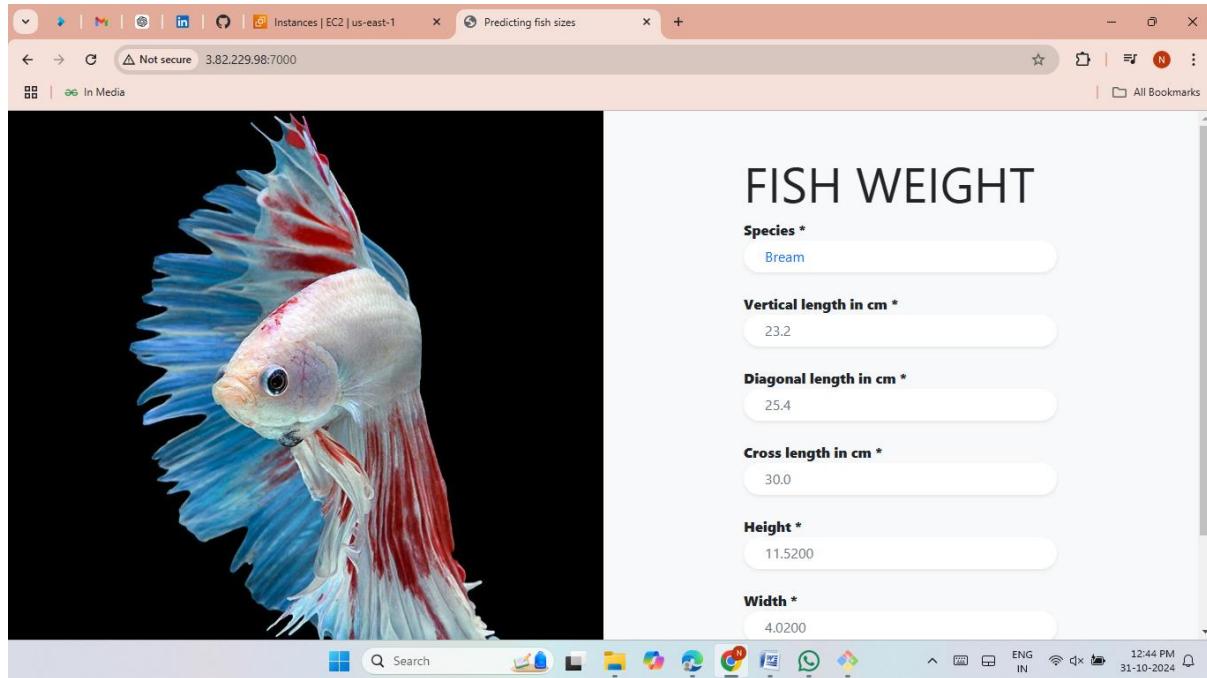
- Here the docker container was created successfully.

```

ec2-user@ip-172-31-83-137:~$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
fish latest 6215702604d1 About a minute ago 1.22GB
[ec2-user@ip-172-31-83-137 ~]$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
[ec2-user@ip-172-31-83-137 ~]$ docker run -dt --name yash -p 7000:8000 fish
db5175ad077e8013eab5987b0c3d2173f901679764628b9874e971f78bd2aa9
[ec2-user@ip-172-31-83-137 ~]$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
db5175ad077e fish "python3 ./app.py" 4 seconds ago Up 3 seconds 9000/tcp, 0.0.0.0:7000->8000/tcp, ::7000->8000/tcp yash
[ec2-user@ip-172-31-83-137 ~]$ |

```

- Now go to EC2 instance and copy public IP and paste it on Google browser along with given port and check the fish python application is displays.



- Now run the following commands to push the docker images into DockerHub platform.

docker login (give the DockerHub user and password).

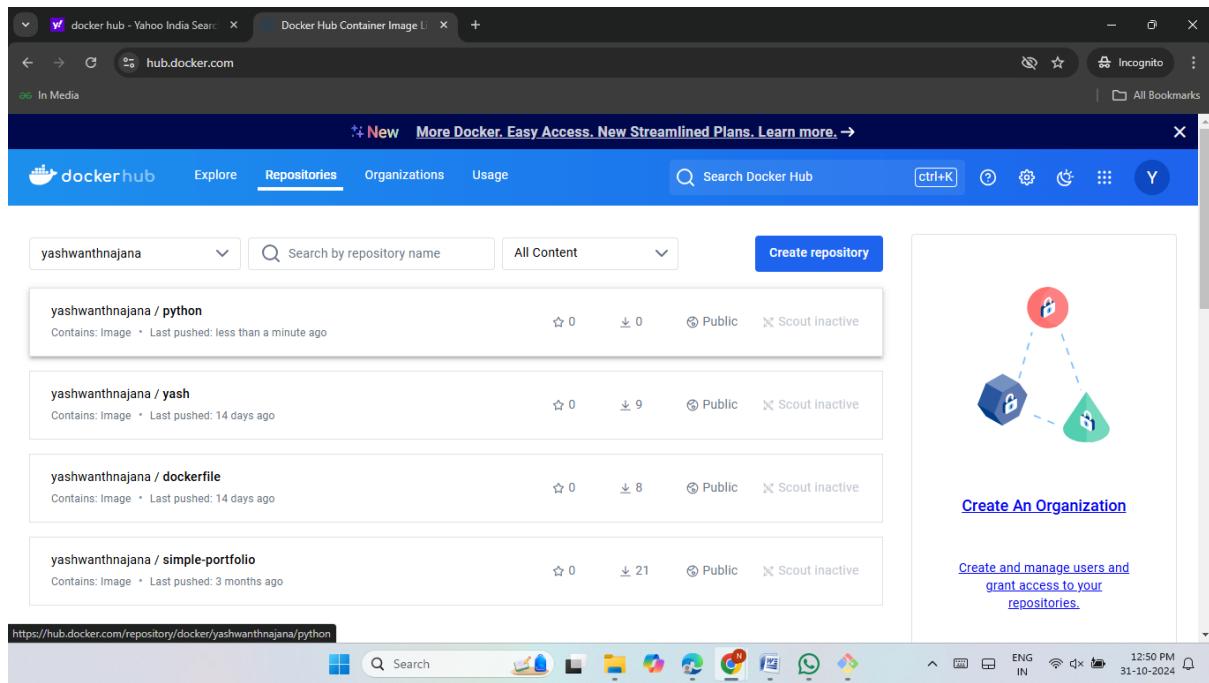
docker tag <image name> <username>/<repo name>:<tag>

docker push <username>/<repo name>:<tag>

```
ec2-user@ip-172-31-83-137:~]$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
fish latest 6215702604d1 2 minutes ago 1.22GB
[ec2-user@ip-172-31-83-137 ~]$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
NAMES
db5175ad077e fish "python3 ./app.py" About a minute ago Up About a minute 9000/tcp, 0.0.0.0:7000->8000/tcp, :::7000->8000/tcp
yash
[ec2-user@ip-172-31-83-137 ~]$ docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/
Username: yashwanthnjana
Password:
WARNING! Your password will be stored unencrypted in /home/ec2-user/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[ec2-user@ip-172-31-83-137 ~]$ docker tag fish yashwanthnjana/python:latest
[ec2-user@ip-172-31-83-137 ~]$ docker push yashwanthnjana/python:latest
The push refers to repository [docker.io/yashwanthnjana/python]
be26665191e4: Pushed
5f70bf18a086: Pushed
eb6b205c3e50: Pushed
c608797282de: Pushed
985c76601b8e: Pushed
fcac1b22fe3d: Mounted from library/amazonlinux
latest: digest: sha256:2d97cc0fd5d152dfc8858acb8e16ce71a1ec77bf91dced8b0dff3de3fccea83a size: 1585
[ec2-user@ip-172-31-83-137 ~]$
```

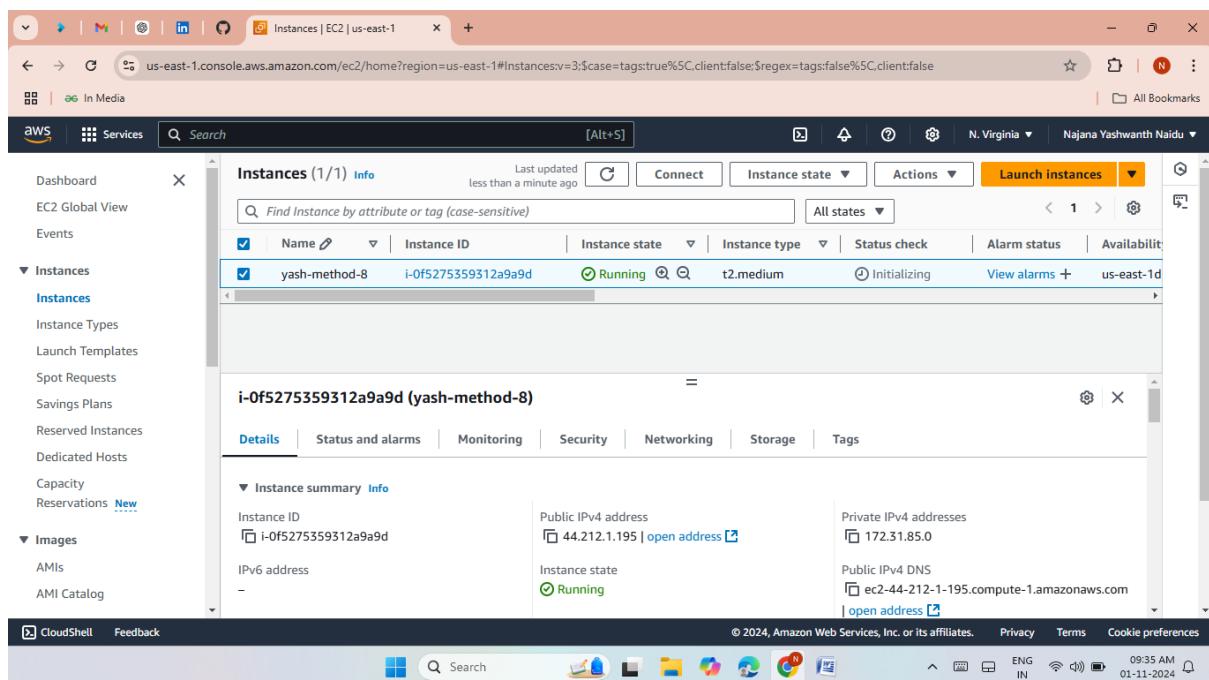
- Here the docker image is pushed successfully (python).



METHOD -8

Build and deploy python applications with Docker and k8's (EKS and KOPS) (use Declarative manifest method along with docker image)

- Now launch an EC2 instance for hosting the fish python application in kubernetes.



- Now install aws cli (in linux by default aws cli is there. If we want updated version, we can download).
- Create role with all admin permissions and attach it to the EC2 instance.
- Now first Install kubectl package from Google.
- Then install kubectl.

```

ec2-user@ip-172-31-85-0:~ yaswa@LAPTOP-GM32QJHO MINGW64 ~/keys
$ ssh -i "yash.pem" ec2-user@ec2-44-212-1-195.compute-1.amazonaws.com
The authenticity of host 'ec2-44-212-1-195.compute-1.amazonaws.com (44.212.1.195)' can't be established.
ED25519 key fingerprint is SHA256:aftmOP8xshvQ3higRXGVonpcjpAofZjsflluEgSNgY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-212-1-195.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

          #_
          ~\###_      Amazon Linux 2
          ~\###\     AL2 End of Life is 2025-06-30.
          ~\#\_
          ~\#\_
          ~\#\_>
          ~\#\_> A newer version of Amazon Linux is available!
          ~\#\_>
          ~\#\_> Amazon Linux 2023, GA and supported until 2028-03-15.
          ~\#\_>
          ~\#\_> https://aws.amazon.com/linux/amazon-linux-2023/
9 package(s) needed for security, out of 12 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-85-0 ~]$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload  Total  Spent   Left  Speed
100  138  100  138    0     0  3014      0 --:--:-- --:--:-- 3066
100 53.7M 100 53.7M    0     0  90.1M      0 --:--:-- --:--:-- 94.3M
[ec2-user@ip-172-31-85-0 ~]$ ls
kubectl
[ec2-user@ip-172-31-85-0 ~]$ chmod +x ./kubectl
[ec2-user@ip-172-31-85-0 ~]$ ls
kubectl
[ec2-user@ip-172-31-85-0 ~]$ sudo mv ./kubectl /usr/local/bin/kubectl
[ec2-user@ip-172-31-85-0 ~]$ ls
[ec2-user@ip-172-31-85-0 ~]$ kubectl version --client
Client Version: v1.31.2
Kustomize Version: v5.4.2
[ec2-user@ip-172-31-85-0 ~]$
```

- Now install kops from Google.

```

ec2-user@ip-172-31-85-0:~ yaswa@LAPTOP-GM32QJHO MINGW64 ~/keys
          #_
          ~\###_      Amazon Linux 2
          ~\###\     AL2 End of Life is 2025-06-30.
          ~\#\_
          ~\#\_
          ~\#\_>
          ~\#\_> A newer version of Amazon Linux is available!
          ~\#\_>
          ~\#\_> Amazon Linux 2023, GA and supported until 2028-03-15.
          ~\#\_>
          ~\#\_> https://aws.amazon.com/linux/amazon-linux-2023/
9 package(s) needed for security, out of 12 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-85-0 ~]$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload  Total  Spent   Left  Speed
100  138  100  138    0     0  3014      0 --:--:-- --:--:-- 3066
100 53.7M 100 53.7M    0     0  90.1M      0 --:--:-- --:--:-- 94.3M
[ec2-user@ip-172-31-85-0 ~]$ ls
kubectl
[ec2-user@ip-172-31-85-0 ~]$ chmod +x ./kubectl
[ec2-user@ip-172-31-85-0 ~]$ ls
kubectl
[ec2-user@ip-172-31-85-0 ~]$ sudo mv ./kubectl /usr/local/bin/kubectl
[ec2-user@ip-172-31-85-0 ~]$ ls
[ec2-user@ip-172-31-85-0 ~]$ kubectl version --client
Client Version: v1.31.2
Kustomize Version: v5.4.2
[ec2-user@ip-172-31-85-0 ~]$ curl -Lo kops https://github.com/kubernetes/kops/releases/download/$(curl -s https://api.github.com/repos/kubernetes/kops/releases/latest | grep tag_name | cut -d '"' -f 4)/kops-linux-amd64
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload  Total  Spent   Left  Speed
  0     0     0     0     0     0      0 --:--:-- --:--:--      0
100 238M 100 238M    0     0  94.5M      0 0:00:02 0:00:02 --:--:-- 99.7M
[ec2-user@ip-172-31-85-0 ~]$ ls
kops
[ec2-user@ip-172-31-85-0 ~]$ chmod +x kops
[ec2-user@ip-172-31-85-0 ~]$ sudo mv kops /usr/local/bin/kops
[ec2-user@ip-172-31-85-0 ~]$ kops version
Client version: 1.30.1 (git-v1.30.1)
[ec2-user@ip-172-31-85-0 ~]$
```

- Now run the following commands.

aws s3 mb s3://yash-k8 (to create s3 bucket to store the data).

aws s3 ls (to show the s3).

```
export KOPS_STATE_STORE=s3://yash-k8 (attach s3 bucket to kops for backup).
```

ssh-keygen (now enter).

```
kops create cluster --name .k8s.local --state s3:// --zones useast-1b,us-east-1a --node-count 2 --yes (to create Cluster and it create VPC, Subnet, IG, Route, LoadBalancer, Target Group ,Auto Scalling Group, Instances).
```

kops validate cluster (to see the details od cluster).

```
ec2-user@ip-172-31-85-0:~$ kops
% Total    % Received % Xferd  Average Speed   Time      Time      Time  Current
                                         Dload  Upload   Total   Spent   Left  Speed
100   0     0       0       0        0      0  ---:---:---:---:---:---:---:--- 0
[ec2-user@ip-172-31-85-0 ~]$ ls
kops
[ec2-user@ip-172-31-85-0 ~]$ chmod +x kops
[ec2-user@ip-172-31-85-0 ~]$ sudo mv kops /usr/local/bin/kops
[ec2-user@ip-172-31-85-0 ~]$ kops version
Client version: 1.30.1 (git-v1.30.1)
[ec2-user@ip-172-31-85-0 ~]$ aws s3 mb s3://yash-k8
make_bucket: yash-k8
[ec2-user@ip-172-31-85-0 ~]$ aws s3 ls
2024-06-12 10:09:13 elasticbeanstalk-us-east-1-637423323663
2024-10-31 11:49:02 yash-k8
[ec2-user@ip-172-31-85-0 ~]$ export KOPS_STATE_STORE=s3://yash-k8
[ec2-user@ip-172-31-85-0 ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ec2-user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ec2-user/.ssh/id_rsa.
Your public key has been saved in /home/ec2-user/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:EfjfzyqrHQYhC0dyCwf1fp6xUM7ebz0kfHcIBqdMU ec2-user@ip-172-31-85-0.ec2.internal
The key's randomart image is:
+---[RSA 2048]----+
| ..oo oo+o. |
| oo =o= oE o |
| .o.*.+ = . |
| .. +o.. + |
| . oS+ + . |
| . B+= + |
| . .oO+ o |
| . . . .# o |
| . . . . . . |
+---[SHA256]----+
[ec2-user@ip-172-31-85-0 ~]$
```

```
ec2-user@ip-172-31-85-0:~
```

NAME	ROLE	READY
i-01353bd1b853f6d03	node	True
i-0c47f0b5eea901aa2	control-plane	True
i-0c8ba35a85b9e4b63	node	True

KIND	NAME	MESSAGE
Pod	kube-system/cilium-2csth	system-node-critical pod "cilium-2csth" is not ready (cilium-agent)
Pod	kube-system/coredns-7bfbb444bf-7rxpc	system-cluster-critical pod "coredns-7bfbb444bf-7rxpc" is pending
Pod	kube-system/coredns-autoscaler-84969b654b-td6nh	system-cluster-critical pod "coredns-autoscaler-84969b654b-td6nh" is pending
Pod	kube-system/ebs-csi-node-4vvcs	system-node-critical pod "ebs-csi-node-4vvcs" is pending
Pod	kube-system/ebs-csi-node-rr/r2	system-node-critical pod "ebs-csi-node-rr/r2" is pending

Validation Failed
Error: validation failed: cluster not yet healthy
[ec2-user@ip-172-31-85-0 ~]\$ kubectl get pods
No resources found in default namespace.
[ec2-user@ip-172-31-85-0 ~]\$ kops validate cluster
Using cluster from kubectl context: yashwanth.k8s.local

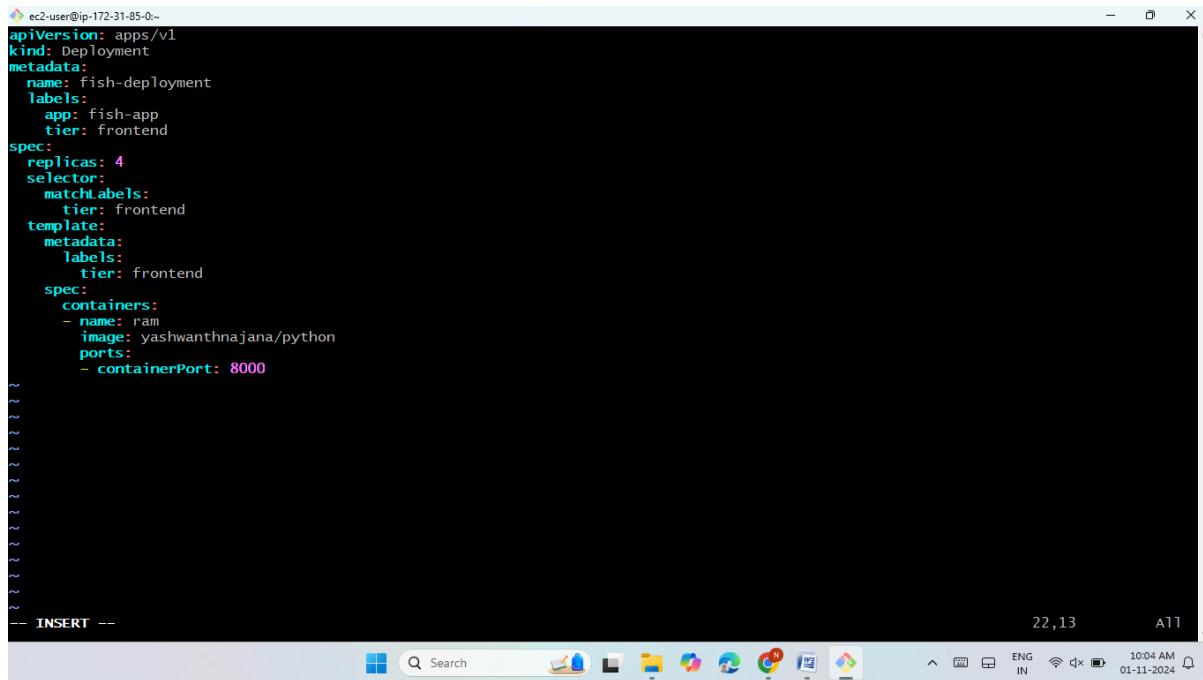
Validating cluster yashwanth.k8s.local

INSTANCE GROUPS					
NAME	ROLE	MACHINETYPE	MIN	MAX	SUBNETS
control-plane-us-east-1a	ControlPlane	t3.medium	1	1	us-east-1a
nodes-us-east-1a	Node	t3.medium	1	1	us-east-1a
nodes-us-east-1d	Node	t3.medium	1	1	us-east-1d

NAME	ROLE	READY
i-01353bd1b853f6d03	node	True
i-0c47f0b5eea901aa2	control-plane	True
i-0c8ba35a85b9e4b63	node	True

Your cluster yashwanth.k8s.local is ready
[ec2-user@ip-172-31-85-0 ~]\$ |

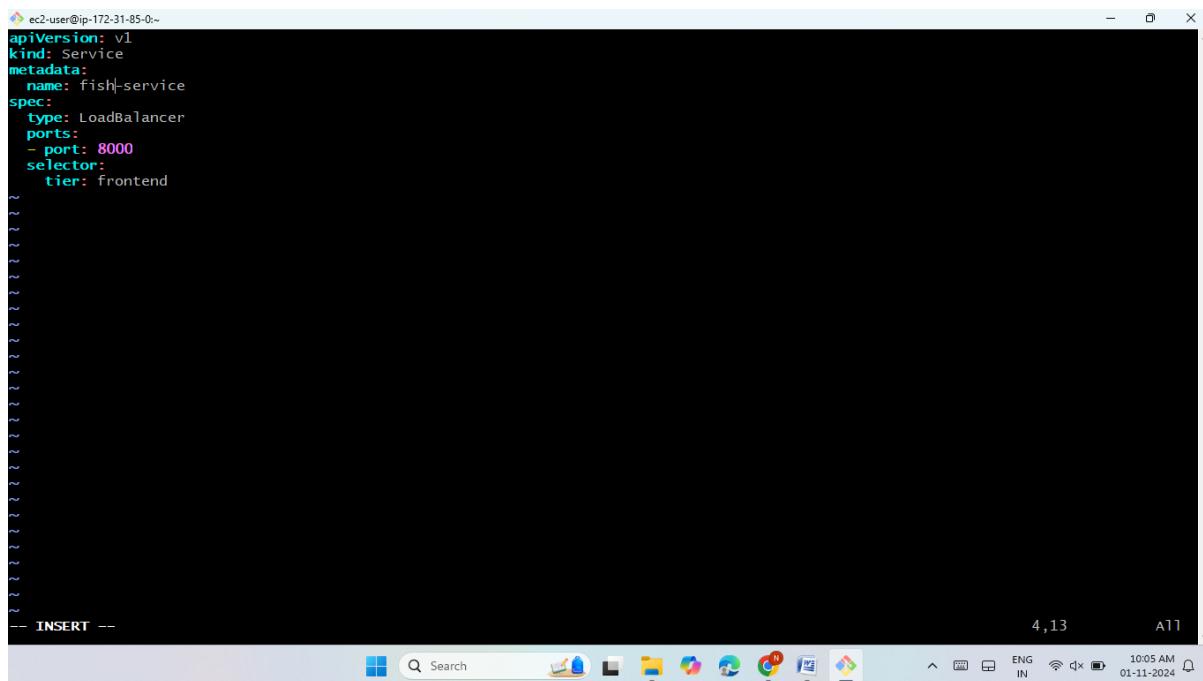
- Now create a deployment.yaml file for run the dockerhub image by using declarative manifest method.



```
ec2-user@ip-172-31-85-0:~$ 
apiVersion: apps/v1
kind: Deployment
metadata:
  name: fish-deployment
  labels:
    app: fish-app
    tier: frontend
spec:
  replicas: 4
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
        - name: ram
          image: yashwanthnajana/python
          ports:
            - containerPort: 8000
-- INSERT --
```

The screenshot shows a terminal window on a Windows operating system. The command `apiVersion: apps/v1` is being typed. The terminal background is black, and the text is white. The status bar at the bottom right shows the date and time as 10:04 AM on 01-11-2024.

- Now create a service.yaml file for run the dockerhub image by using declarative manifest method.



```
ec2-user@ip-172-31-85-0:~$ 
apiVersion: v1
kind: Service
metadata:
  name: fish-service
spec:
  type: LoadBalancer
  ports:
    - port: 8000
  selector:
    tier: frontend
-- INSERT --
```

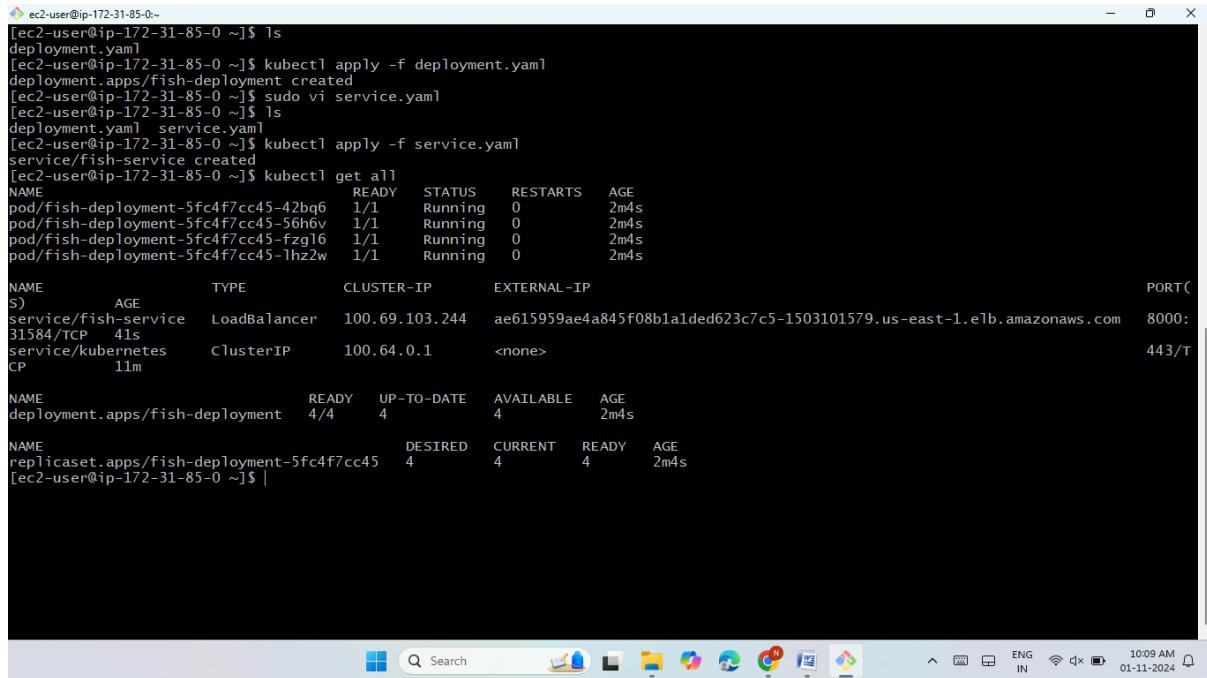
The screenshot shows a terminal window on a Windows operating system. The command `apiVersion: v1` is being typed. The terminal background is black, and the text is white. The status bar at the bottom right shows the date and time as 10:05 AM on 01-11-2024.

- Now run the following command.

```
kubectl apply -f deployment.yaml
```

```
kubectl apply -f service.yaml
```

```
kubectl get all
```



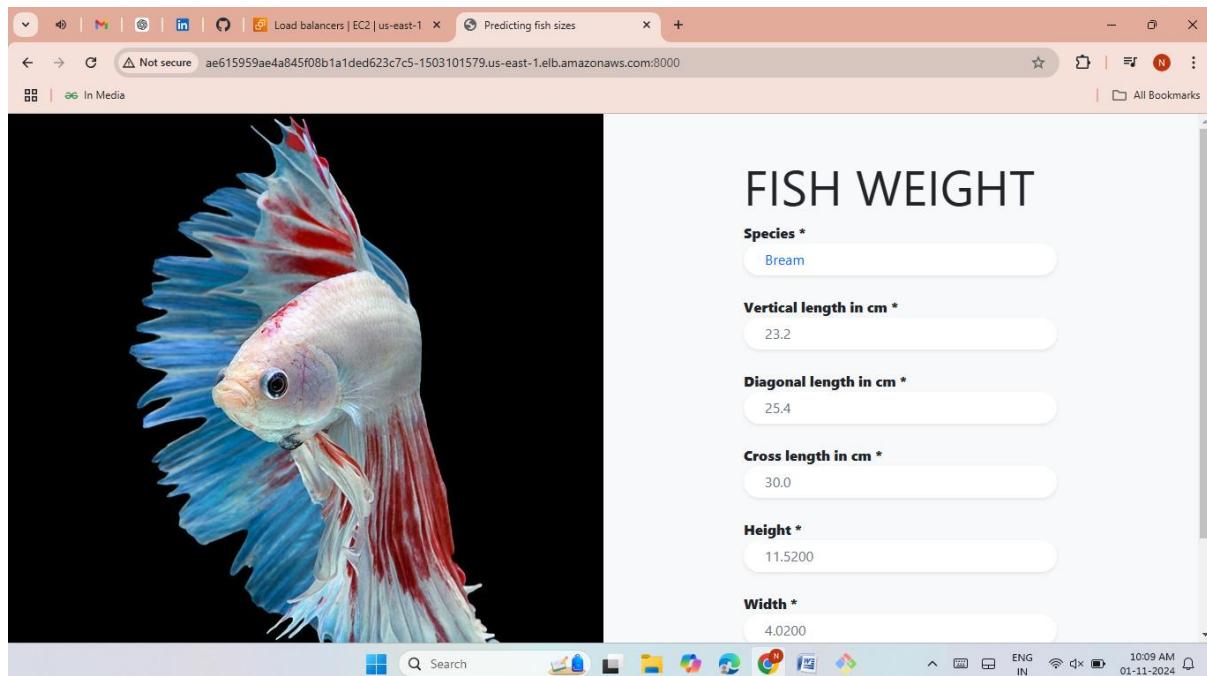
```
[ec2-user@ip-172-31-85-0 ~]$ ls
deployment.yaml
[ec2-user@ip-172-31-85-0 ~]$ kubectl apply -f deployment.yaml
deployment.apps/fish-deployment created
[ec2-user@ip-172-31-85-0 ~]$ sudo vi service.yaml
[ec2-user@ip-172-31-85-0 ~]$ ls
deployment.yaml service.yaml
[ec2-user@ip-172-31-85-0 ~]$ kubectl apply -f service.yaml
service/fish-service created
[ec2-user@ip-172-31-85-0 ~]$ kubectl get all
NAME           READY   STATUS    RESTARTS   AGE
pod/fish-deployment-5fc4f7cc45-42bg6   1/1     Running   0          2m4s
pod/fish-deployment-5fc4f7cc45-56h6v   1/1     Running   0          2m4s
pod/fish-deployment-5fc4f7cc45-fzg16   1/1     Running   0          2m4s
pod/fish-deployment-5fc4f7cc45-1hz2w   1/1     Running   0          2m4s

NAME              TYPE        CLUSTER-IP      EXTERNAL-IP
SVC             AGE
service/fish-service   LoadBalancer   100.69.103.244   ae615959ae4a845f08b1a1ded623c7c5-1503101579.us-east-1.elb.amazonaws.com   8000:31584/TCP
41s
service/kubernetes   ClusterIP    100.64.0.1      <none>
443/TCP
11m

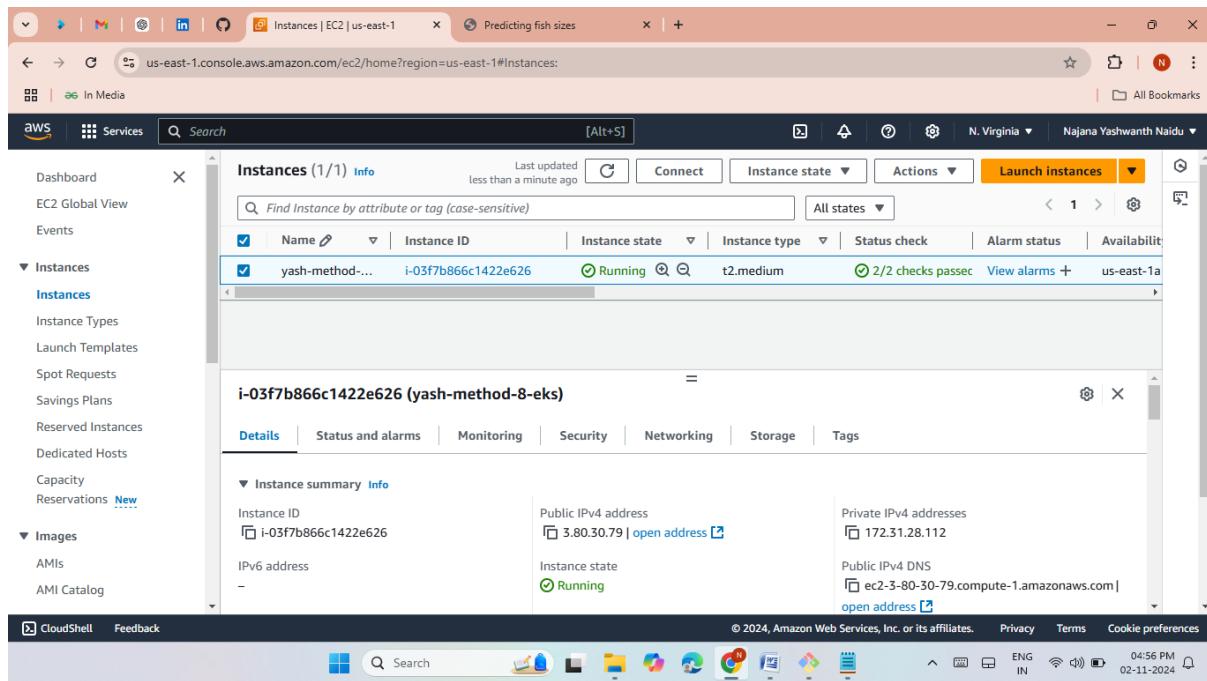
NAME            READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/fish-deployment   4/4       4           4          2m4s

NAME           DESIRED   CURRENT   READY   AGE
replicaset.apps/fish-deployment-5fc4f7cc45   4         4         4          2m4s
[ec2-user@ip-172-31-85-0 ~]$ |
```

- Now with help of Load Balancer DNS link along with given port the fish python we application was hosted successfully.



- Now launch an EC2 instance for hosting the fish python application in kubernetes.



- Now install aws cli (in linux by default aws cli is there. If we want updated version, we can download).
- Now first Install kubectl package from Google.
- Then install kubectl.
- Then Install eksctl (CLI tool for creating and managing EKS clusters).

```
ec2-user@ip-172-31-28-112:~$ 
Dependencies Resolved

=====
Package          Arch      Version       Repository      Size
=====
Installing:
  kubectl        x86_64    1.31.2-150500.1.1   kubernetes    11 M
Transaction Summary
=====
Install 1 Package

Total download size: 11 M
Installed size: 54 M
Downloading packages:
warning: /var/cache/yum/x86_64/2/kubernetes/packages/kubectl-1.31.2-150500.1.1.x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID 9a296436: NOKEY
Public key for kubectl-1.31.2-150500.1.1.x86_64.rpm is not installed
kubectl-1.31.2-150500.1.1.x86_64.rpm
Retrieving key from https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
Importing GPG key 0x9a296436:
  Userid  : "jsv:kubernetes OBS Project <jsv:kubernetes@build.opensuse.org>"
  Fingerprint: de15 b144 86cd 377b 9e87 6e1a 2346 54da 9a29 6436
  From    : https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : kubectl-1.31.2-150500.1.1.x86_64
  Verifying   : kubectl-1.31.2-150500.1.1.x86_64
1/1
1/1

Installed:
  kubectl.x86_64 0:1.31.2-150500.1.1

Complete!
```

```

[ec2-user@ip-172-31-28-112 ~]$ ls
[ec2-user@ip-172-31-28-112 ~]$ curl -o eksctl https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz
[ec2-user@ip-172-31-28-112 ~]$ ls
[ec2-user@ip-172-31-28-112 ~]$ tar -xzf eksctl_$(uname -s)_amd64.tar.gz
tar (child): eksctl_linux_amd64.tar.gz: Cannot open: No such file or directory
tar (child): Error is not recoverable: exiting now
tar: Child returned status 2
tar: Error is not recoverable: exiting now
[ec2-user@ip-172-31-28-112 ~]$ sudo mv eksctl /usr/local/bin
[ec2-user@ip-172-31-28-112 ~]$ eksctl version
-bash: /usr/local/bin/eksctl: Permission denied
[ec2-user@ip-172-31-28-112 ~]$ chmod +x ./eksctl
chmod: cannot access './eksctl': No such file or directory
[ec2-user@ip-172-31-28-112 ~]$ sudo chmod +x /usr/local/bin/eksctl
[ec2-user@ip-172-31-28-112 ~]$ eksctl version
[ec2-user@ip-172-31-28-112 ~]$ sudo yum remove awscli
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
--> Package awscli.noarch 0:1.18.147-1.amzn2.0.2 will be erased
--> Finished Dependency Resolution
amzn2-core/2/x86_64
Dependencies Resolved

=====
| 3.6 kB 00:00:00
=====

```

Package	Arch	Version	Repository	Size
awscli	noarch	1.18.147-1.amzn2.0.2	installed	7.9 M

05:01 PM
02-11-2024

- Now run the following command.

aws configure

Aws access keys

Aws secret keys

Region

Output format

- Now create cluster with the command.

eksctl create cluster --name <cluster name> --region <region> --nodegroup-name <nodegroup name> --nodes <count>

```

[ec2-user@ip-172-31-28-112 ~]$ eksctl create cluster --name yash1-cluster --region us-east-1 --nodegroup-name yash1-nodes --nodes 3
2024-11-02 11:07:49 [0] eksctl version 0.194.0
2024-11-02 11:07:49 [0] using region us-east-1
2024-11-02 11:07:49 [0] setting availability zones to [us-east-1c us-east-1d]
2024-11-02 11:07:49 [0] subnets for us-east-1c - public:192.168.0.0/19 private:192.168.64.0/19
2024-11-02 11:07:49 [0] subnets for us-east-1d - public:192.168.32.0/19 private:192.168.96.0/19
2024-11-02 11:07:49 [0] nodegroup "yash1-nodes" will use "" [AmazonLinux2/1.30]
2024-11-02 11:07:49 [0] using Kubernetes version 1.30
2024-11-02 11:07:49 [0] creating EKS cluster "yash1-cluster" in "us-east-1" region with managed nodes
2024-11-02 11:07:49 [0] will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup
2024-11-02 11:07:49 [0] if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=us-east-1 --cluster=yash1-cluster'
2024-11-02 11:07:49 [0] Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "yash1-cluster" in "us-east-1"
2024-11-02 11:07:49 [0] Cloudwatch logging will not be enabled for cluster "yash1-cluster" in "us-east-1"
2024-11-02 11:07:49 [0] you can enable it with 'eksctl utils update-cluster-logging --enable-types=[SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)] --region=us-east-1 --cluster=yash1-cluster'
2024-11-02 11:07:49 [0] default addons vpc-cni, kube-proxy, coredns were not specified, will install them as EKS addons
2024-11-02 11:07:49 [0] 2 sequential tasks: { create cluster control plane "yash1-cluster",
2 sequential sub-tasks: {
2 sequential sub-tasks: {
1 task: { create addons },
wait for control plane to become ready,
},
create managed nodegroup "yash1-nodes",
}
}
2024-11-02 11:07:49 [0] building cluster stack "eksctl-yash1-cluster-cluster"
2024-11-02 11:07:50 [0] deploying stack "eksctl-yash1-cluster-cluster"
2024-11-02 11:08:20 [0] waiting for CloudFormation stack "eksctl-yash1-cluster-cluster"
2024-11-02 11:08:50 [0] waiting for CloudFormation stack "eksctl-yash1-cluster-cluster"
2024-11-02 11:09:50 [0] waiting for CloudFormation stack "eksctl-yash1-cluster-cluster"
2024-11-02 11:10:50 [0] waiting for CloudFormation stack "eksctl-yash1-cluster-cluster"
2024-11-02 11:11:50 [0] waiting for CloudFormation stack "eksctl-yash1-cluster-cluster"

```

04:57 PM
02-11-2024

```

ec2-user@ip-172-31-28-112:~ 
2024-11-02 11:13:50 [o] waiting for CloudFormation stack "eksctl-yashi1-cluster-cluster"
2024-11-02 11:14:50 [o] waiting for CloudFormation stack "eksctl-yashi1-cluster-cluster"
2024-11-02 11:15:50 [o] waiting for CloudFormation stack "eksctl-yashi1-cluster-cluster"
2024-11-02 11:15:51 [!] recommended policies were found for "vpc-cni" addon, but since OIDC is disabled on the cluster, eksctl cannot configure the requested permissions; the recommended way to provide IAM permissions for "vpc-cni" addon is via pod identity associations; after addon creation is completed, add all recommended policies to the config file, under 'addon.PodIdentityAssociations', and run 'eksctl update addon'
2024-11-02 11:15:51 [o] creating addon
2024-11-02 11:15:51 [o] successfully created addon
2024-11-02 11:15:51 [o] creating addon
2024-11-02 11:15:52 [o] successfully created addon
2024-11-02 11:15:52 [o] creating addon
2024-11-02 11:15:52 [o] successfully created addon
2024-11-02 11:17:52 [o] building managed nodegroup stack "eksctl-yashi1-cluster-nodegroup-yashi1-nodes"
2024-11-02 11:17:53 [o] deploying stack "eksctl-yashi1-cluster-nodegroup-yashi1-nodes"
2024-11-02 11:17:53 [o] waiting for CloudFormation stack "eksctl-yashi1-cluster-nodegroup-yashi1-nodes"
2024-11-02 11:18:23 [o] waiting for CloudFormation stack "eksctl-yashi1-cluster-nodegroup-yashi1-nodes"
2024-11-02 11:19:10 [o] waiting for CloudFormation stack "eksctl-yashi1-cluster-nodegroup-yashi1-nodes"
2024-11-02 11:20:20 [o] waiting for CloudFormation stack "eksctl-yashi1-cluster-nodegroup-yashi1-nodes"
2024-11-02 11:20:20 [o] waiting for the control plane to become ready
2024-11-02 11:20:21 [✓] saved kubeconfig as "/home/ec2-user/.kube/config"
2024-11-02 11:20:21 [o] no tasks
2024-11-02 11:20:21 [✓] all EKS cluster resources for "yashi1-cluster" have been created
2024-11-02 11:20:21 [✓] created 0 nodegroup(s) in cluster "yashi1-cluster"
2024-11-02 11:20:21 [o] nodegroup "yashi1-nodes" has 3 node(s)
2024-11-02 11:20:21 [o] node "ip-192-168-31-154.ec2.internal" is ready
2024-11-02 11:20:21 [o] node "ip-192-168-33-145.ec2.internal" is ready
2024-11-02 11:20:21 [o] node "ip-192-168-8-239.ec2.internal" is ready
2024-11-02 11:20:21 [o] waiting for at least 3 node(s) to become ready in "yashi1-nodes"
2024-11-02 11:20:21 [o] nodegroup "yashi1-nodes" has 3 node(s)
2024-11-02 11:20:21 [o] node "ip-192-168-31-154.ec2.internal" is ready
2024-11-02 11:20:21 [o] node "ip-192-168-33-145.ec2.internal" is ready
2024-11-02 11:20:21 [o] node "ip-192-168-8-239.ec2.internal" is ready
2024-11-02 11:20:21 [✓] created 1 managed nodegroup(s) in cluster "yashi1-cluster"
2024-11-02 11:20:22 [o] kubectl command should work with "/home/ec2-user/.kube/config", try 'kubectl get nodes'
2024-11-02 11:20:22 [✓] EKS cluster "yashi1-cluster" in "us-east-1" region is ready

```

- After the cluster is created, configure kubectl to use the new cluster with the command
aws eks --region <region> update-kubeconfig --name <cluster name>
- Now create a deployment.yaml file for run the dockerhub image by using declarative manifest method.
- Now create a service.yaml file for run the dockerhub image by using declarative manifest method.
- Now run the following command.
kubectl apply -f deployment.yaml
kubectl apply -f service.yaml
kubectl get all

```

ec2-user@ip-172-31-28-112:~ 
2024-11-02 11:20:21 [o] node "ip-192-168-8-239.ec2.internal" is ready
2024-11-02 11:20:21 [✓] created 1 managed nodegroup(s) in cluster "yashi1-cluster"
2024-11-02 11:20:22 [o] kubectl command should work with "/home/ec2-user/.kube/config", try 'kubectl get nodes'
2024-11-02 11:20:22 [✓] EKS cluster "yashi1-cluster" in "us-east-1" region is ready
[ec2-user@ip-172-31-28-112 ~]$ aws eks --region us-east-1 update-kubeconfig --name yashi1-cluster
Added new context arn:aws:eks:us-east-1:637423323663:cluster/yashi1-cluster to /home/ec2-user/.kube/config
[ec2-user@ip-172-31-28-112 ~]$ kubectl get all
NAME           TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
service/kubernetes   ClusterIP  10.100.0.1   <none>        443/TCP   8m52s
[ec2-user@ip-172-31-28-112 ~]$ sudo vi deployment.yaml
[ec2-user@ip-172-31-28-112 ~]$ kubectl apply -f deployment.yaml
deployment.apps/fish-deployment created
[ec2-user@ip-172-31-28-112 ~]$ sudo vi service.yaml
[ec2-user@ip-172-31-28-112 ~]$ kubectl apply -f service.yaml
service/fish-service created
[ec2-user@ip-172-31-28-112 ~]$ kubectl get all
NAME           READY   STATUS    RESTARTS   AGE
pod/fish-deployment-865d6dbdc4-2lqdh 1/1    Running   0          53s
pod/fish-deployment-865d6dbdc4-cj7wz  1/1    Running   0          53s
pod/fish-deployment-865d6dbdc4-hm29c  1/1    Running   0          53s
pod/fish-deployment-865d6dbdc4-wlxgl  1/1    Running   0          53s
NAME           AGE     TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)
service/fish-service   LoadBalancer  10.100.36.110  a474745dc4fb64ccc84c4ebd2404bb4a-772218851.us-east-1.elb.amazonaws.com  8000:31
service/kubernetes   ClusterIP  10.100.0.1   <none>        443/TCP
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/fish-deployment  4/4      4           4          53s
NAME           DESIRED  CURRENT   READY   AGE
replicaset.apps/fish-deployment-865d6dbdc4  4       4        4      53s
[ec2-user@ip-172-31-28-112 ~]$ ^C
[ec2-user@ip-172-31-28-112 ~]$ 

```

```
ec2-user@ip-172-31-28-112:~  
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  name: fish-deployment  
  labels:  
    app: fish-app  
    tier: frontend  
spec:  
  replicas: 4  
  selector:  
    matchLabels:  
      tier: frontend  
template:  
  metadata:  
    labels:  
      tier: frontend  
  spec:  
    containers:  
    - name: yashwanth  
      image: yashwanthnajana/python  
      ports:  
      - containerPort: 8000  
  
-- INSERT --
```

- Here the AWS EKS Cluster was created.

The screenshot shows the AWS EKS Clusters console. On the left, there's a sidebar with 'Amazon Elastic Kubernetes Service' and 'Clusters'. The main area displays a table titled 'Clusters (1) Info' with one row. The row contains the cluster name 'yash1-cluster', status 'Active', Kubernetes version '1.30', support period 'Standard support until July 28, 2025', and upgrade policy 'Extended'. There are buttons for 'Edit', 'Delete', and 'Add cluster' at the top of the table. The bottom of the screen shows the Windows taskbar with various pinned icons.

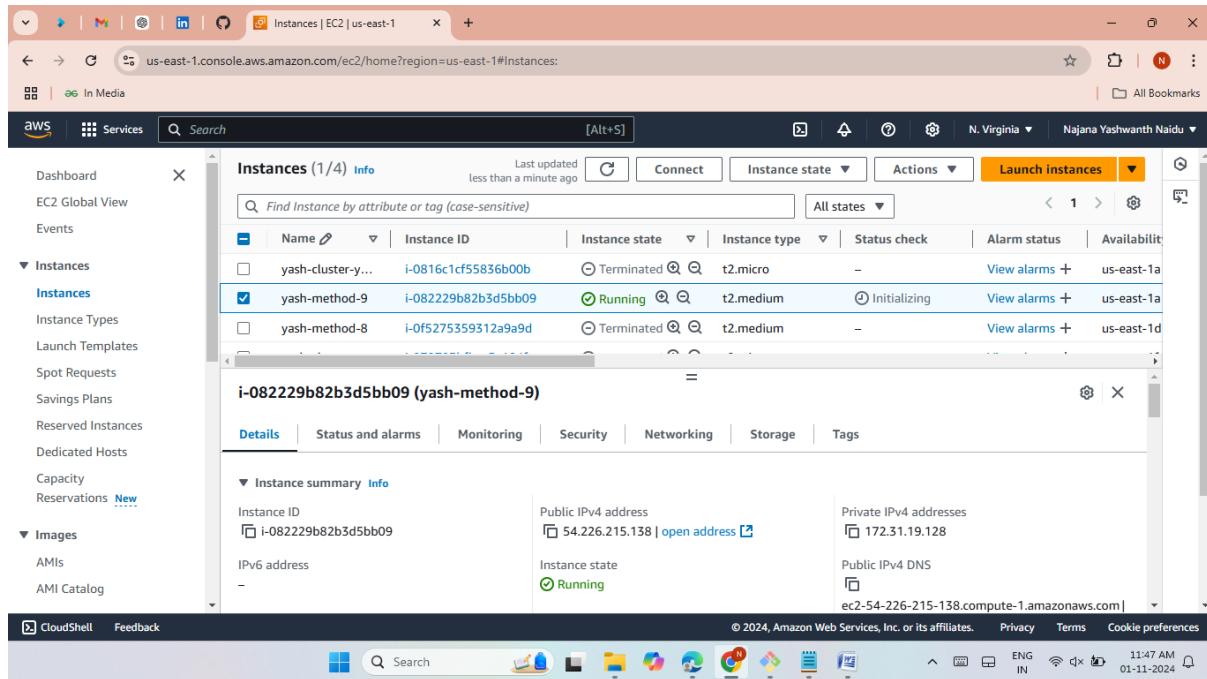
- Now with help of Load Balancer DNS link along with given port the fish python we application was hosted successfully.

The screenshot shows a web browser window. The address bar indicates the URL is 'Instances | EC2 | us-east-1' and the page content is 'Predicting fish sizes'. The page features a large image of a Siamese fighting fish (Betta splendens) on the left. To the right, there's a form titled 'FISH WEIGHT' with fields for 'Species *' (set to 'Bream'), 'Vertical length in cm *' (23.2), 'Diagonal length in cm *' (25.4), 'Cross length in cm *' (30.0), 'Height *' (11.5200), and 'Width *' (4.0200). The bottom of the screen shows the Windows taskbar.

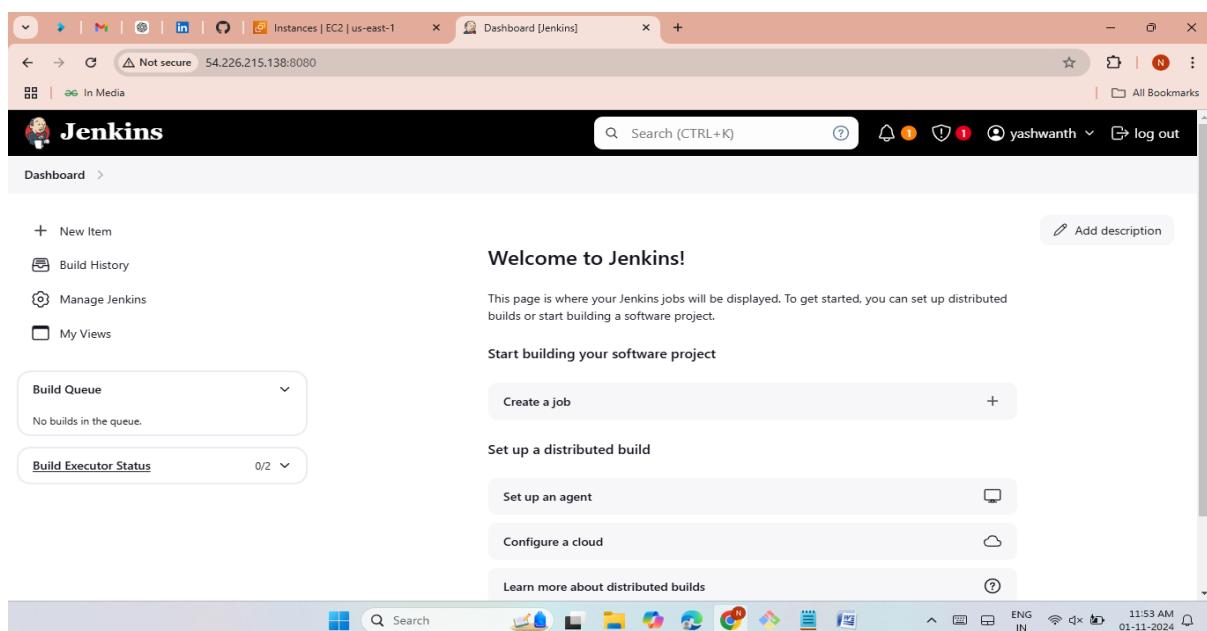
METHOD-9

Build and deploy python applications with the Git, github, Jenkins and Terraform (with Build periodically, pollscm and webhooks)

- First launch the EC2 instance with the Jenkins port number(8080) and required ports.



- Now Install git and Jenkins.
- Now start and enable the Jenkins.
- Here the Jenkins was launched successfully.



➤ Now install the AWS credentials plugin.

The screenshot shows the Jenkins Plugin Manager interface. On the left, there's a sidebar with options like 'Updates', 'Available plugins' (which is selected), 'Installed plugins', 'Advanced settings', and 'Download progress'. The main area has a search bar at the top with 'aws' typed in. Below it is a table with columns for 'Install', 'Name', and 'Released'. There are four entries:

- AWS Credentials** 231.v08a_59f17d742: Allows storing Amazon IAM credentials within the Jenkins Credentials API. Store Amazon IAM access keys (AWSAccessKeyId and AWSSecretKey) within the Jenkins Credentials API. Also support IAM Roles and IAM MFA Token. Released 7 mo 7 days ago.
- Amazon Web Services SDK :: Minimal** 1.12.772-474.v7ff79a_2046a_fb_: Library plugins (for use by other plugins) aws. Minimal modules for the AWS SDK for Java. Released 17 days ago.
- Amazon Web Services SDK :: EC2** 1.12.772-474.v7ff79a_2046a_fb_: Library plugins (for use by other plugins) aws. EC2 module for the AWS SDK for Java. Released 17 days ago.
- Amazon Web Services SDK :: All** 1.12.772-474.v7ff79a_2046a_fb_: Library plugins (for use by other plugins) aws. All modules for the AWS SDK for Java. Released 17 days ago.

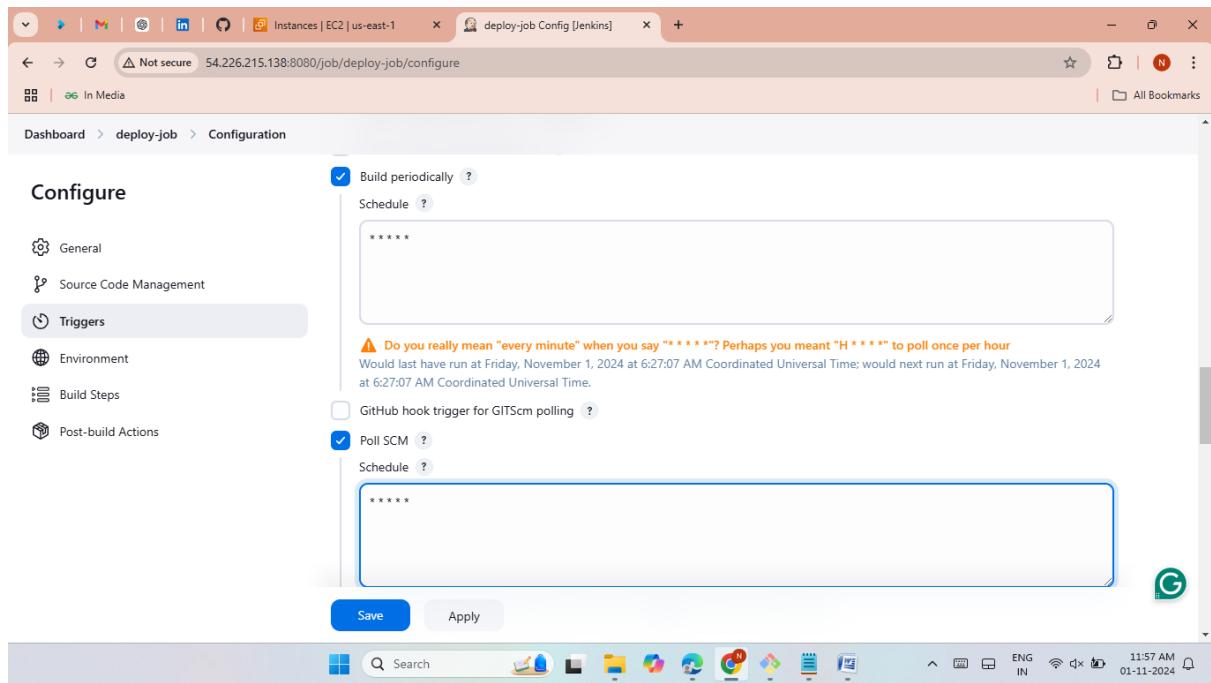
➤ Now create a job and clone the terraform script form github.

The screenshot shows the Jenkins job configuration page for 'deploy-job'. On the left, there's a sidebar with 'General', 'Source Code Management' (selected), 'Triggers', 'Environment', 'Build Steps', and 'Post-build Actions'. The main area is titled 'Source Code Management' and contains the following fields:

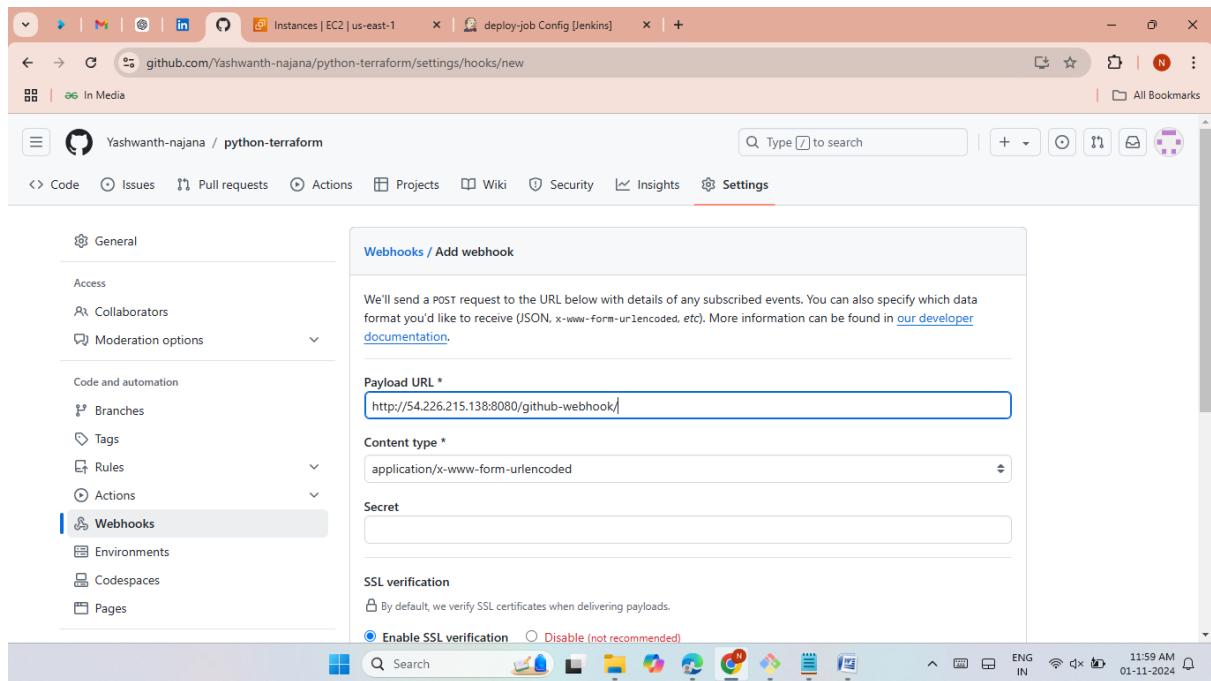
- Repository URL**: https://github.com/Yashwanth-najana/python-terraform.git
- Credentials**: - none - (with a '+ Add' button)

At the bottom are 'Save' and 'Apply' buttons. The status bar at the bottom right shows '11:55 AM 01-11-2024'.

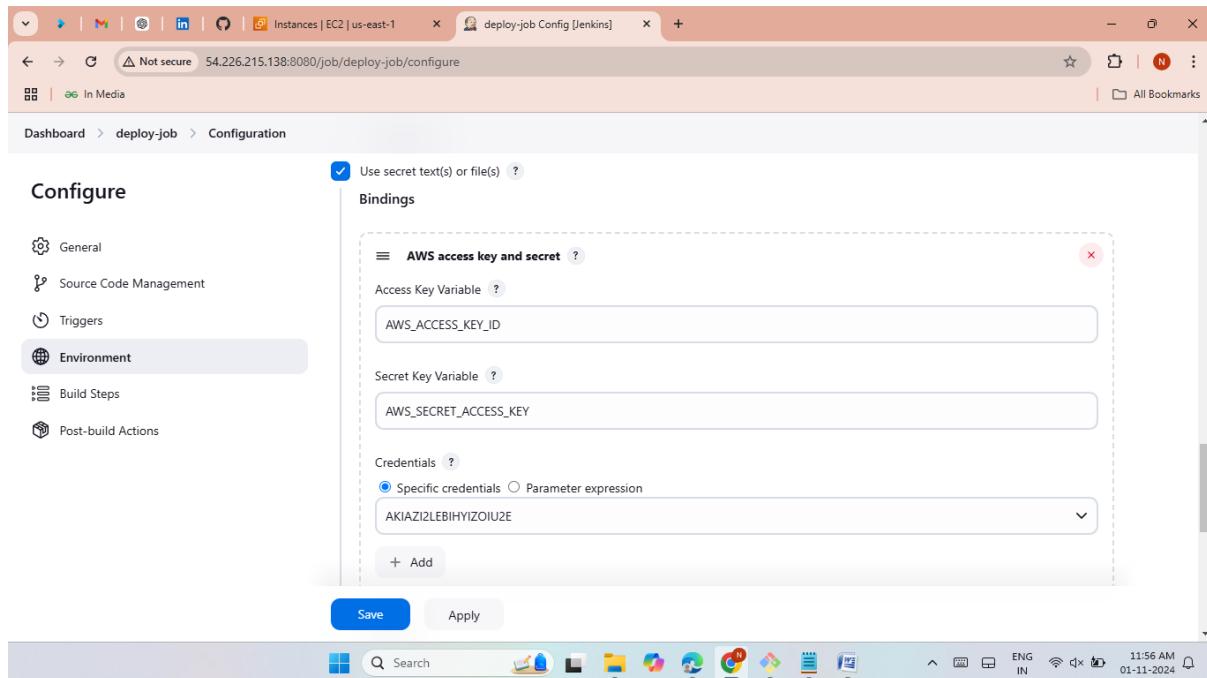
- Now Add the Build Periodically and Poll SCM.



- Create webhooks.
- Github → repo → settings → webhooks → add webhooks → under payload → ip-adress with Jenkins port/github-webhook/



- Then create the AWS credentials.
- Now connect the access keys and secret keys.



- Now install the terraform in execute shell.
- Now add the terraform commands.

```
#!/bin/bash

sudo yum install -y yum-utils shadow-utils

sudo yum-config-manager --add-repo
https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo

sudo yum -y install terraform

sudo chmod + data.sh

terraform init

terraform fmt

terraform validate

terraform plan

terraform apply --auto-approve
```

The screenshot shows the Jenkins job configuration interface. On the left, a sidebar lists options: General, Source Code Management, Triggers, Environment, Build Steps (which is selected and highlighted in grey), and Post-build Actions. The main area is titled "Build Steps" with the sub-instruction "Automate your build process with ordered tasks like code compilation, testing, and deployment." It contains a single step named "Execute shell". The command field contains the following Terraform deployment script:

```

#!/bin/bash
sudo yum install -y yum-utils shadow-utils
sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
sudo yum -y install terraform
sudo chmod +x data.sh
terraform init
terraform fmt
terraform validate
terraform apply --auto-approve

```

At the bottom of the screen, the Windows taskbar is visible with various icons and the system tray showing the date and time.

- Give permissions to jenkins.

Sudo visudo

jenkins ALL=(ALL) NOPASSWD: ALL

- Now restart the Jenkins with the command (**sudo systemctl restart Jenkins**).
- Now click on build the job was success.
- It runs every MINUTE HOUR DOM(DAY OF MONTH) MONTH DOW(DAY OF WEEK).
- Because I have given (* * * * *).

The screenshot shows the Jenkins dashboard. On the left, a sidebar menu includes Workspace, Build Now (which is selected and highlighted in blue), Configure, Delete Project, Git Polling Log, and Rename. The main content area has two sections: "Permalinks" which lists recent builds, and "Builds" which shows a list of completed builds for today. The "Builds" section includes a "Filter" input field and a table with columns for build number, timestamp, and status (indicated by green checkmarks). At the bottom of the screen, the Windows taskbar is visible with various icons and the system tray showing the date and time.

- Here the instances was automatically launched every one min.
- Also create the VPC, Subnet, Internet Gateway, Route Table, Security Groups with help of terraform script.
- Because of Build Periodically, Poll SCM and Webhooks.

The screenshot shows the AWS Management Console interface for the EC2 service. The left sidebar is collapsed, and the main content area displays a table titled "Instances (6) Info". The table lists six instances, with the first three being "Running" and the others "Initializing". The columns include Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability. A "Launch instances" button is visible at the top right of the table area. Below the table, a message says "Select an instance". The bottom of the screen shows the standard Windows taskbar with various pinned icons.

- Here the all fish python applications was launced.

The screenshot shows a web browser window with a form for predicting fish sizes. On the left, there is a large, vibrant image of a Siamese fighting fish (Betta splendens) against a black background. To the right of the image, the form has the title "FISH WEIGHT". It includes several input fields with placeholder text and numerical values:

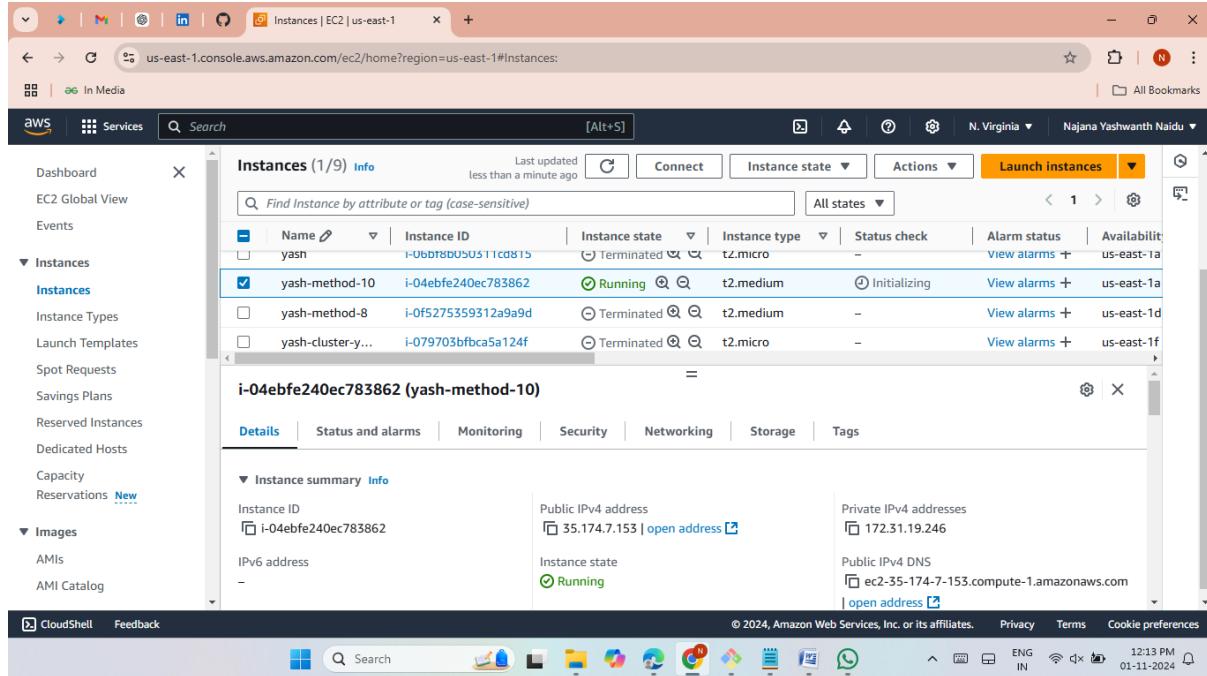
- Species ***: Bream
- Vertical length in cm ***: 23.2
- Diagonal length in cm ***: 25.4
- Cross length in cm ***: 30.0
- Height ***: 11.5200
- Width ***: 4.0200

The browser's address bar shows the URL "34.234.67.226:8000". The bottom of the screen shows the Windows taskbar with various pinned icons.

METHOD-10

Build and deploy python applications with pipeline method (create clone job and Build job) with Build periodically, pollscm and webhooks

- First launch the EC2 instance with the Jenkins port number(8080) and required ports.



- Now Install git and Jenkins.
- Now start and enable the Jenkins.

```
ec2-user@ip-172-31-19-246:~$ Installed size: 92 M
Is this ok [y/N]: y
Downloading packages:
jenkins-2.483-1.1.noarch.rpm
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : jenkins-2.483-1.1.noarch
  Verifying   : jenkins-2.483-1.1.noarch
Installed:
  jenkins.noarch 0:2.483-1.1

Complete!
[ec2-user@ip-172-31-19-246 ~]$ sudo systemctl start jenkins
[ec2-user@ip-172-31-19-246 ~]$ sudo systemctl enable jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service to /usr/lib/systemd/system/jenkins.service.
[ec2-user@ip-172-31-19-246 ~]$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
  Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; vendor preset: disabled)
  Active: active (running) since Fri 2024-11-01 06:46:36 UTC; 13s ago
    Main PID: 10397 (java)
      CGroup: /system.slice/jenkins.service
              └─10397 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080...

Nov 01 06:46:33 ip-172-31-19-246.ec2.internal jenkins[10397]: ****
Nov 01 06:46:33 ip-172-31-19-246.ec2.internal jenkins[10397]: ****
Nov 01 06:46:33 ip-172-31-19-246.ec2.internal jenkins[10397]: ****
Nov 01 06:46:36 ip-172-31-19-246.ec2.internal jenkins[10397]: 2024-11-01 06:46:36.504+0000 [id=49]      INFO      h.m.Down...aller
Nov 01 06:46:36 ip-172-31-19-246.ec2.internal jenkins[10397]: 2024-11-01 06:46:36.504+0000 [id=49]      INFO      hudson.u...#1
Nov 01 06:46:36 ip-172-31-19-246.ec2.internal jenkins[10397]: 2024-11-01 06:46:36.533+0000 [id=33]      INFO      jenkins....ation
Nov 01 06:46:36 ip-172-31-19-246.ec2.internal jenkins[10397]: 2024-11-01 06:46:36.545+0000 [id=23]      INFO      hudson.l...nning
Nov 01 06:46:36 ip-172-31-19-246.ec2.internal systemd[1]: Started Jenkins Continuous Integration Server.
Nov 01 06:46:43 ip-172-31-19-246.ec2.internal systemd[1]: [/usr/lib/systemd/system/jenkins.service:16] unknown lvalue 'StartLi...Unit'
Nov 01 06:46:43 ip-172-31-19-246.ec2.internal systemd[1]: [/usr/lib/systemd/system/jenkins.service:17] unknown lvalue 'StartLi...Unit'
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-19-246 ~]$
```

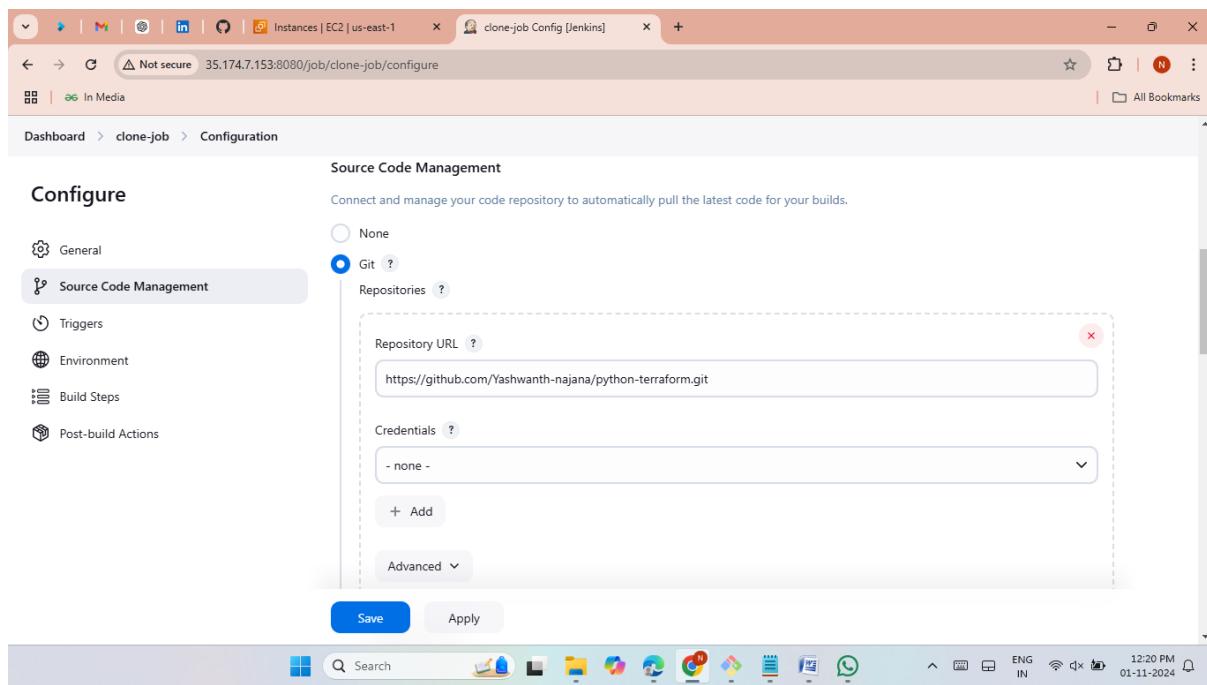
➤ Here the Jenkins was launched successfully.

The screenshot shows the Jenkins dashboard at <http://35.174.7.153:8080>. The main header says "Dashboard [Jenkins]". The left sidebar has links for "New Item", "Build History", "Manage Jenkins", and "My Views". The central area has a "Welcome to Jenkins!" message and a "Start building your software project" section. It includes a "Build Queue" (empty), a "Create a job" button, and sections for "Set up a distributed build", "Set up an agent", "Configure a cloud", and "Learn more about distributed builds". The bottom status bar shows the date and time as 01-11-2024 12:18 PM.

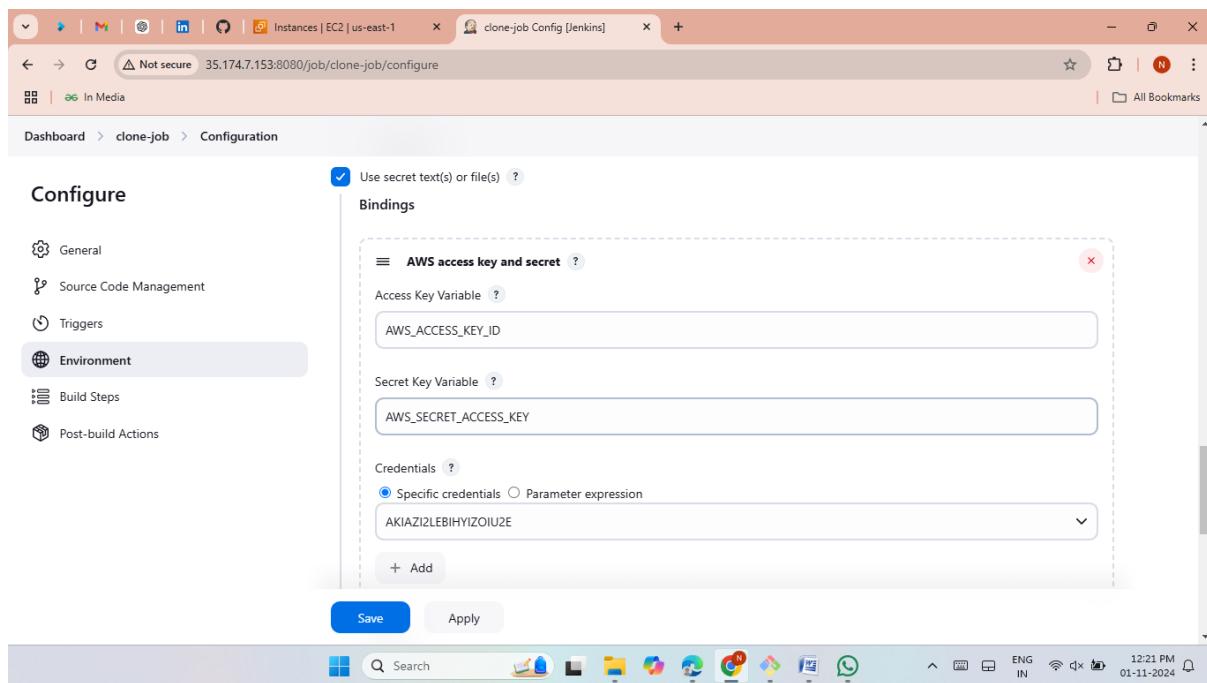
➤ Now install the AWS credentials plugin.

The screenshot shows the Jenkins plugin manager at <http://35.174.7.153:8080/manage/pluginManager/available>. The left sidebar shows "Available plugins" selected. The main area lists the "AWS Credentials" plugin by "aws" with version 231.v08a_59f17d742, which is checked for installation. Other listed plugins include "Amazon Web Services SDK :: Minimal" and "Amazon Web Services SDK :: EC2". The bottom status bar shows the date and time as 01-11-2024 12:19 PM.

- Now create a clone-job and clone the terraform script form github.



- Then create the AWS credentials.
- Now connect the access keys and secret keys.
- Now apply the changes.



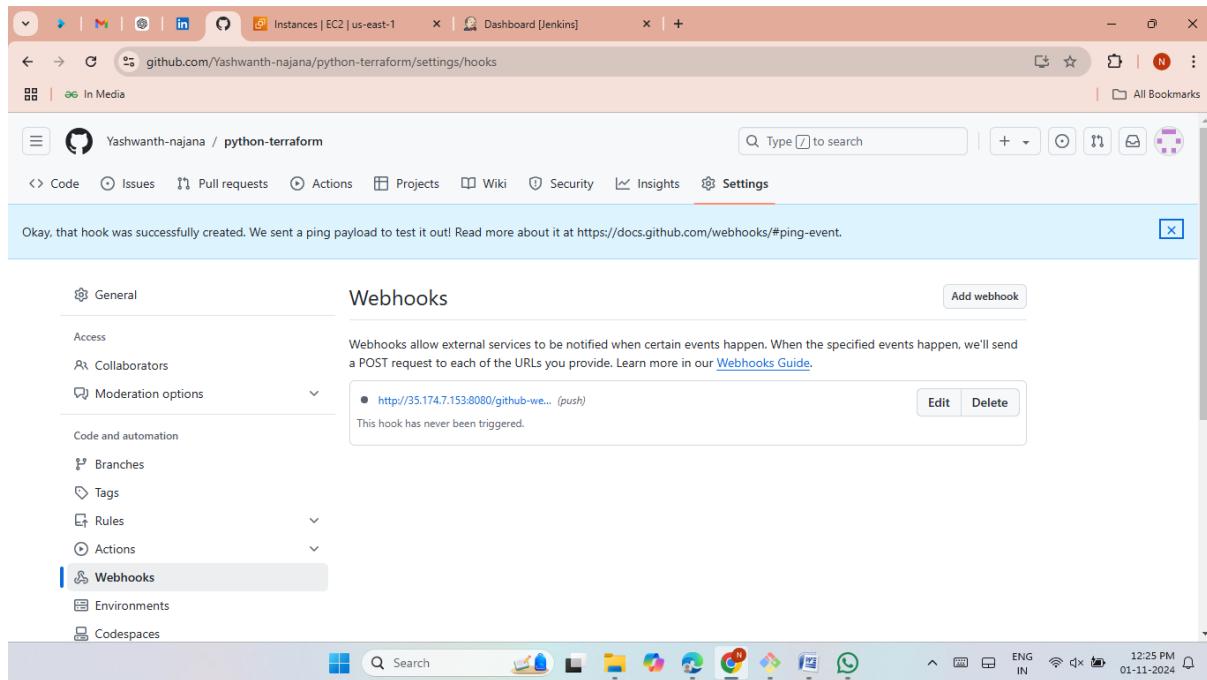
- Here the clone job is success.

The screenshot shows the Jenkins interface for the 'clone-job' project. The top navigation bar indicates the URL is 35.174.7.153:8080/job/clone-job/. The main content area shows the job name 'clone-job' and its description: 'clone the terraform script and deploy python application'. Below this is a 'Permalinks' section. On the left, there's a sidebar with options like 'Status', 'Changes', 'Workspace', 'Build Now' (which is highlighted with a red box), 'Configure', 'Delete Project', and 'Rename'. Under 'Builds', it shows a single build entry for today at 6:52 am, labeled '#1'. The bottom of the screen shows a Windows taskbar with various icons and system status.

- Now create new job and connect to the clone-job.
- Now Add the Build Periodically and Poll SCM.

The screenshot shows the Jenkins interface for the 'deploy-job' configuration page. The top navigation bar indicates the URL is 35.174.7.153:8080/job/deploy-job/configure. The left sidebar shows 'Configure' and several tabs: General, Source Code Management, Triggers (selected), Environment, Build Steps, and Post-build Actions. In the main content area, under 'Triggers', 'Build periodically' is checked with a cron schedule set to '*****'. A warning message states: '⚠ Do you really mean "every minute" when you say "*****"? Perhaps you meant "H * * * *" to poll once per hour'. Below it, 'Poll SCM' is also checked with a cron schedule set to '*****'. At the bottom are 'Save' and 'Apply' buttons. The bottom of the screen shows a Windows taskbar with various icons and system status.

- Now Create Webhooks.
- Github → repo → settings → webhooks → add webhooks → under payload → ip-address with Jenkins port/github-webhook/



- Now install the terraform in execute shell.
- Now add the terraform commands.

```

#!/bin/bash

sudo yum install -y yum-utils shadow-utils

sudo yum-config-manager --add-repo
https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo

sudo yum -y install terraform

sudo chmod +x data.sh

terraform init

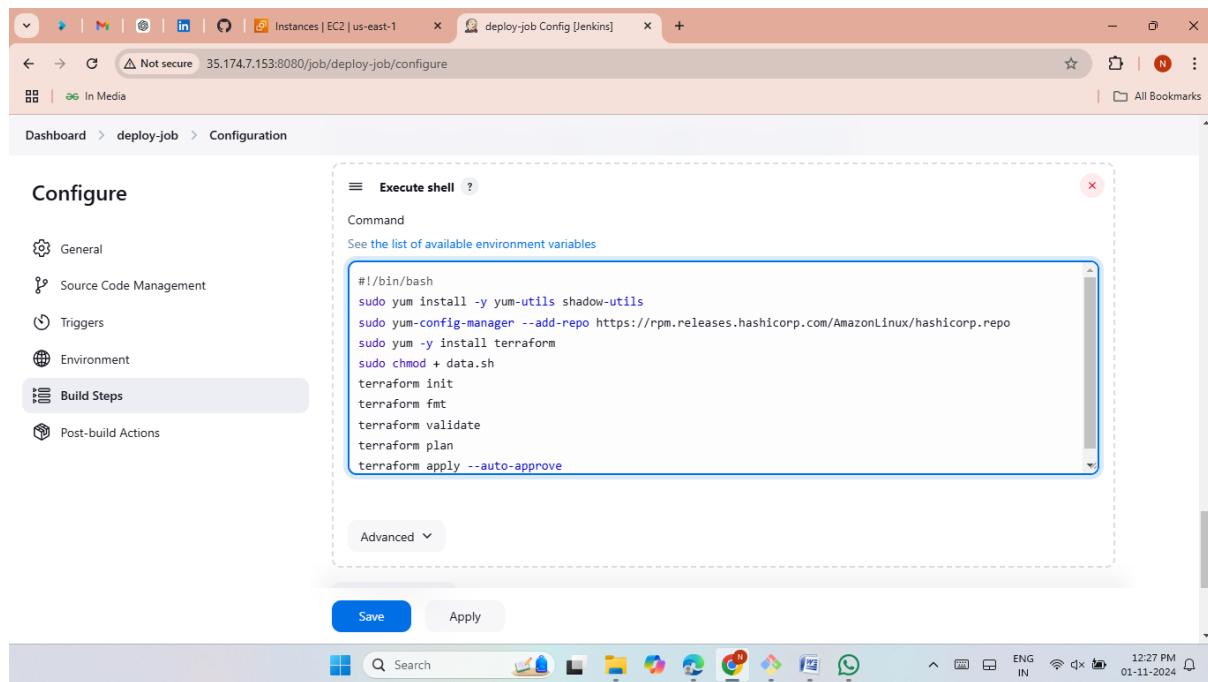
terraform fmt

terraform validate

terraform plan

terraform apply --auto-approve

```

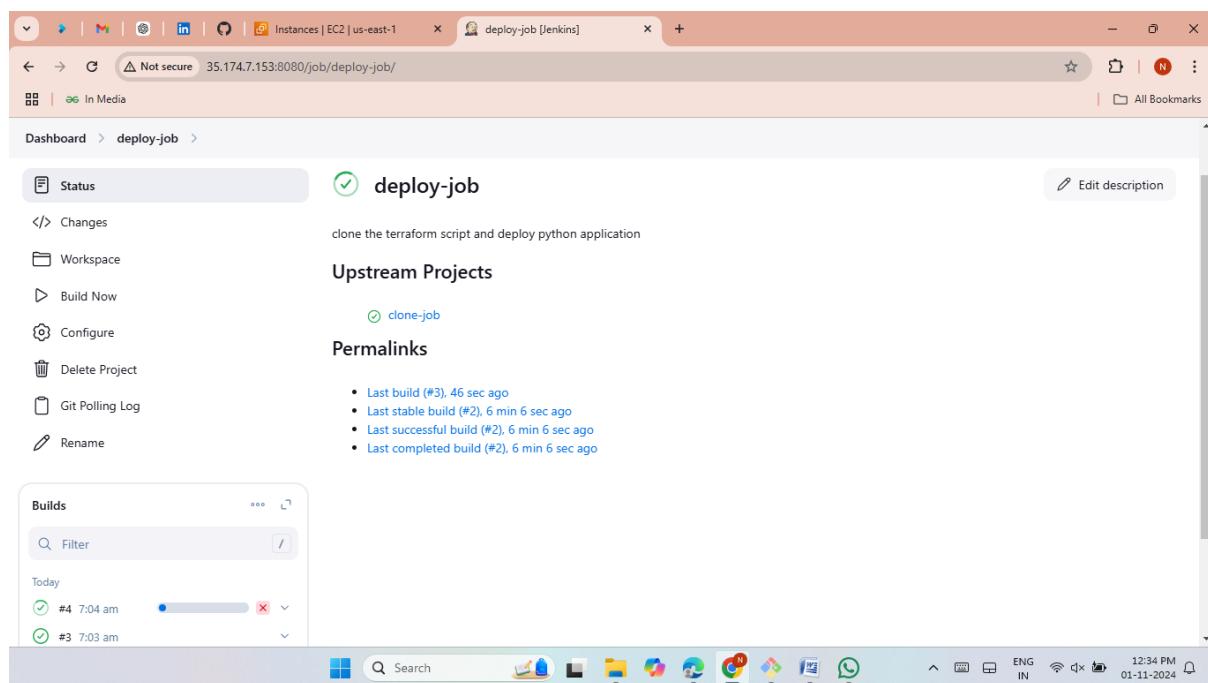


- Give permissions to jenkins.

Sudo visudo

jenkins ALL=(ALL) NOPASSWD: ALL

- Now restart the Jenkins with the command (**sudo systemctl restart Jenkins**).
- Now click on build the job was success.
- It runs every MINUTE HOUR DOM(DAY OF MONTH) MONTH DOW(DAY OF WEEK).
- Because I have given (* * * * *).



- Here the instances was automatically launched every one min.
- Also create the VPC, Subnet, Internet Gateway, Route Table, Security Groups with help of terraform script.
- Because of Build Periodically, Poll SCM and Webhooks.

The screenshot shows the AWS CloudShell interface with the EC2 Instances page open. The left sidebar shows navigation options like Dashboard, EC2 Global View, Events, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, and Images. The main pane shows a table of instances with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability. One instance, 'yash' (Instance ID i-0704f43069dbfb2a0), is selected and highlighted. A modal window for this instance is open, displaying its details: Instance ID i-0704f43069dbfb2a0, Public IPv4 address 18.234.219.74, and Instance state Running. Other tabs in the modal include Status and alarms, Monitoring, Security, Networking, Storage, and Tags.

- Here the all fish python applications was launced.

The screenshot shows a web browser window with a Python application for predicting fish sizes. On the left, there is a large, vibrant image of a Siamese fighting fish (Betta) against a black background. To the right, there is a form with fields for inputting fish dimensions. The fields are labeled: Species * (Bream), Vertical length in cm * (23.2), Diagonal length in cm * (25.4), Cross length in cm * (30.0), Height * (11.5200), and Width * (4.0200). The browser interface includes a search bar, a taskbar with various icons, and a status bar at the bottom indicating the date and time.

- Now install the build pipeline plugin for create a Jenkins pipeline.

Available plugins - Plugins [jenkins]

Search (CTRL+K)

Install

Plugins

Build Pipeline 2.0.2

User Interface Build Tools Other Post-Build Actions

This plugin renders upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins.

Warning: This plugin version may not be safe to use. Please review the following security notices:

- Stored XSS vulnerability

5 mo 17 days ago

This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.

- Now create a Jenkins pipeline.

New view [Jenkins]

Search (CTRL+K)

New view

Name: yash-pipeline

Type: Build Pipeline View

Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.

List View

Shows items in a simple list format. You can choose which jobs are to be displayed in which view.

My View

This view automatically displays all the jobs that the current user has an access to.

Create

- Here the build pipeline is successfully created.

