How to deploy Django Application in AWS ECS

Introduction to Django

Django is a python framework for developing and building web application in the python. It supports raid development and scalability, It follows MVS architecture for developing web apps.

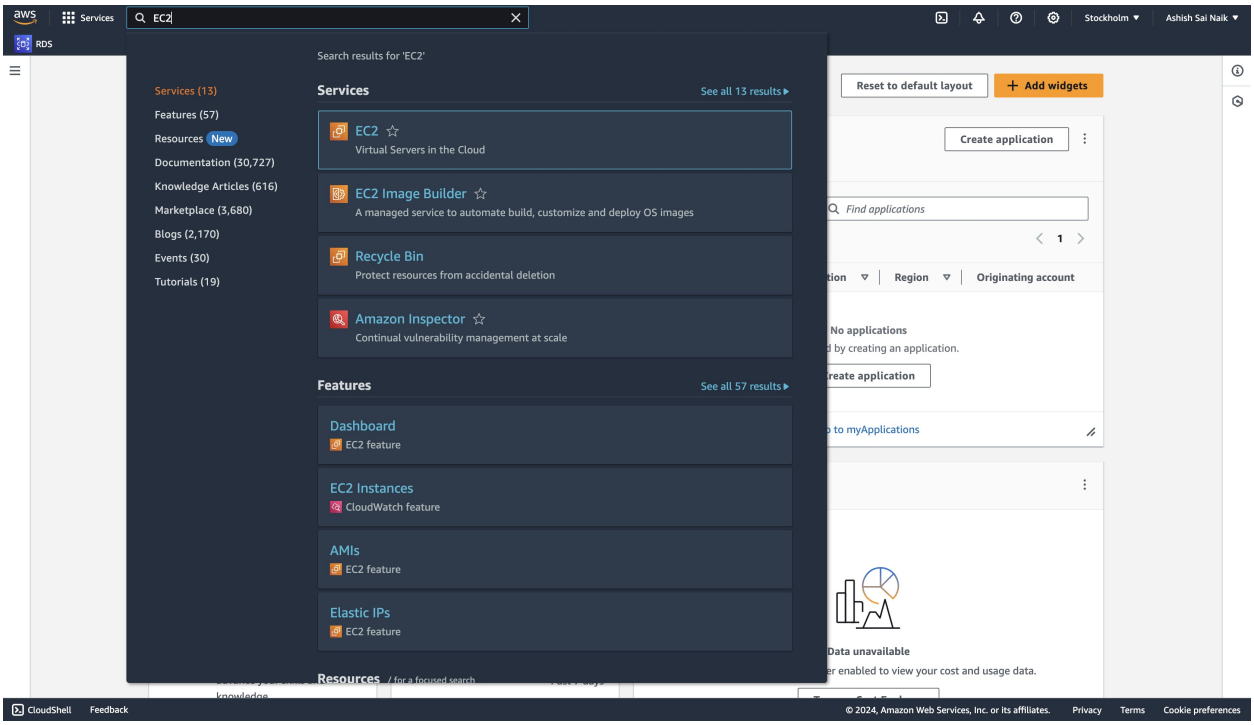
Creating EC2 Instance

Prerequisites

- An active AWS account.
- AWS Management Console login credentials (username and password, or access to an IAM user with the necessary permissions).

1. **Login your AWS account and open EC2 Console**

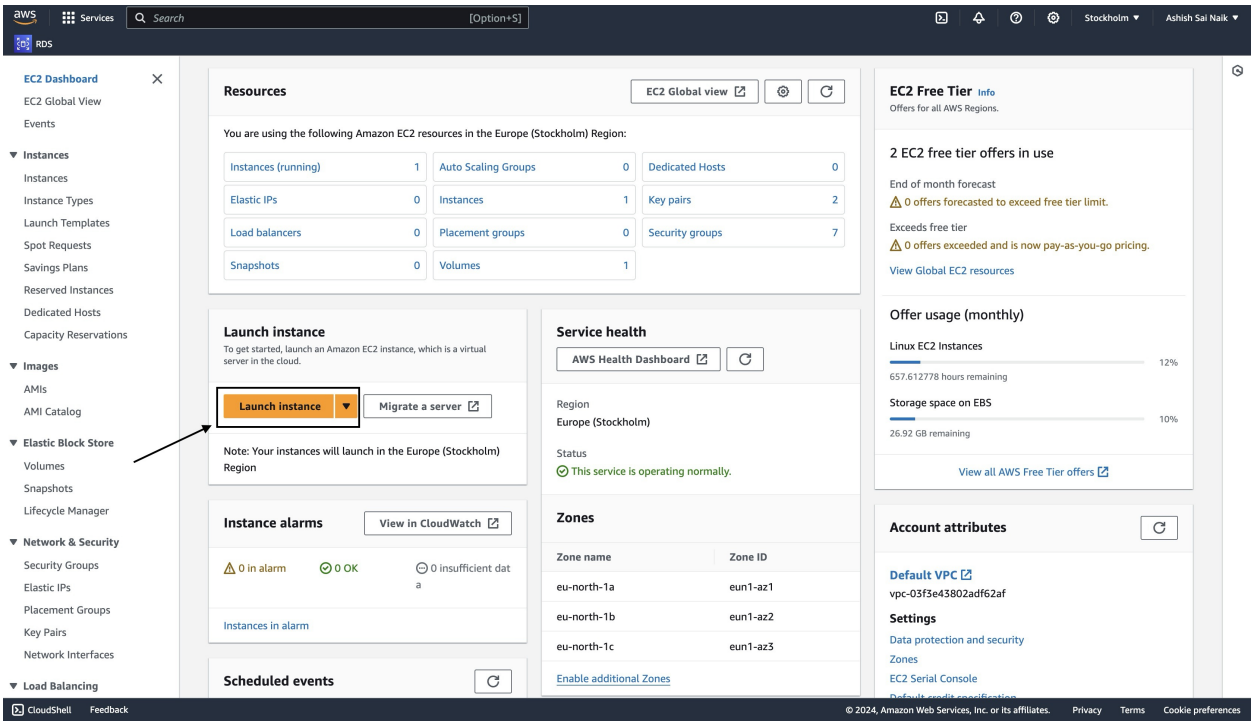
once logged in , you will be on the AWS Management Console home page. from the home page, locate he “search” at the top of the home page, In the "Search" , you can type EC2 , click on "EC2”



Once you Clicked on the EC2, You will taken to the EC2 Dashboard,

• Here, you can manage your EC2 instances, create new instances, configure security groups, manage elastic IPs, and perform other EC2-related tasks.

2. Click Launch Instance



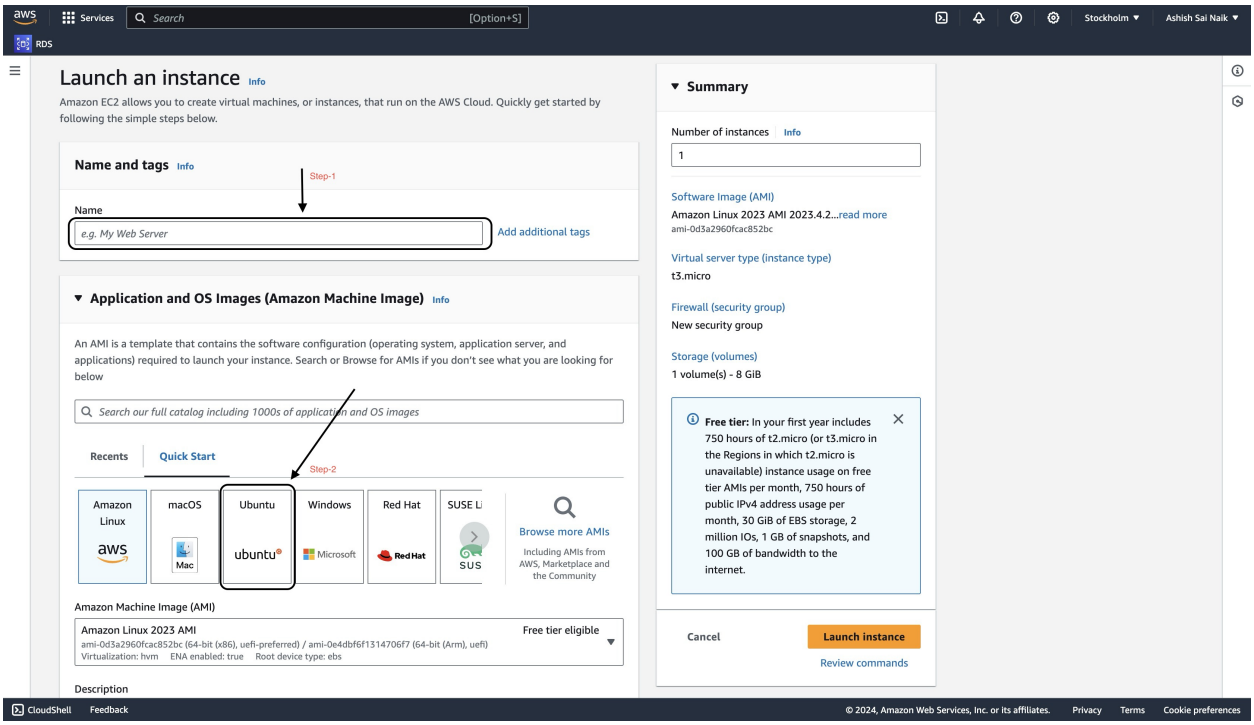
• In the EC2 Dashboard, click on the "Launch Instance" button.
• Follow the steps in the instance launch wizard to configure and launch your new instance.

Step-1

- Once you entered in the Launch an instance, You have to Enter the Name and tag for the Instance

Step-2

- Then once you completed the step-1 make sure opt the Ubuntu for the application and os image as shown in the below fig.



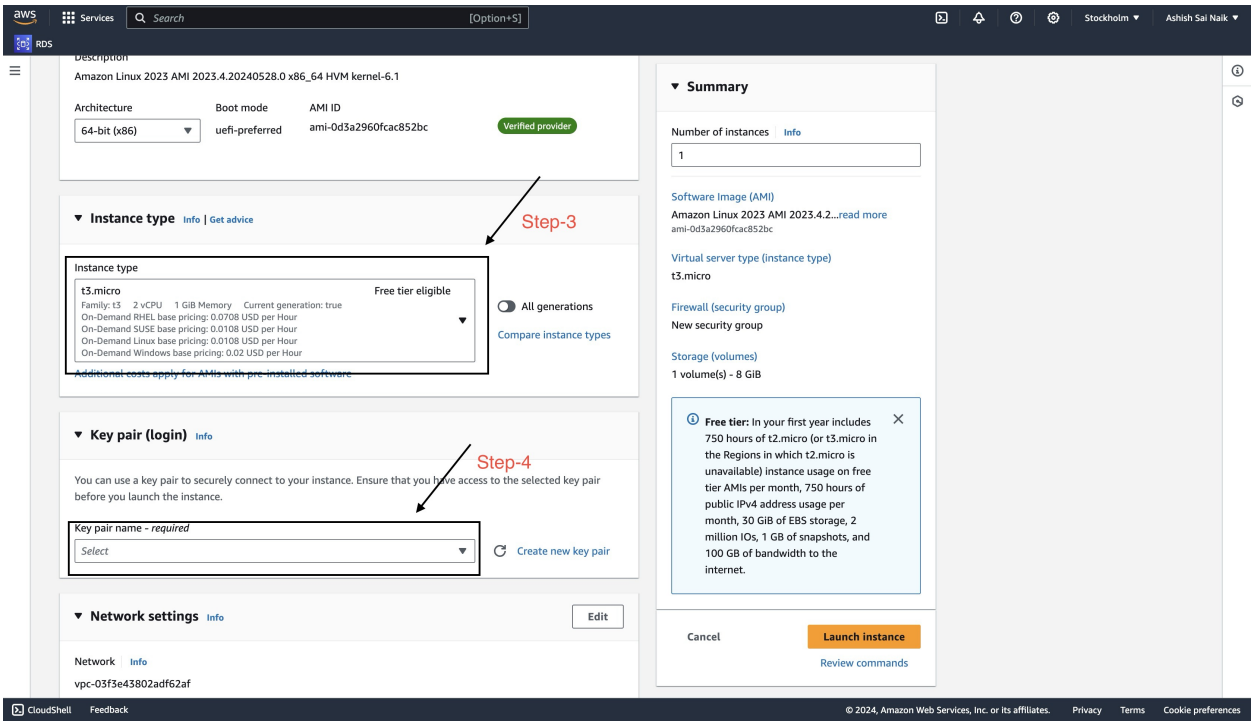
Step-3

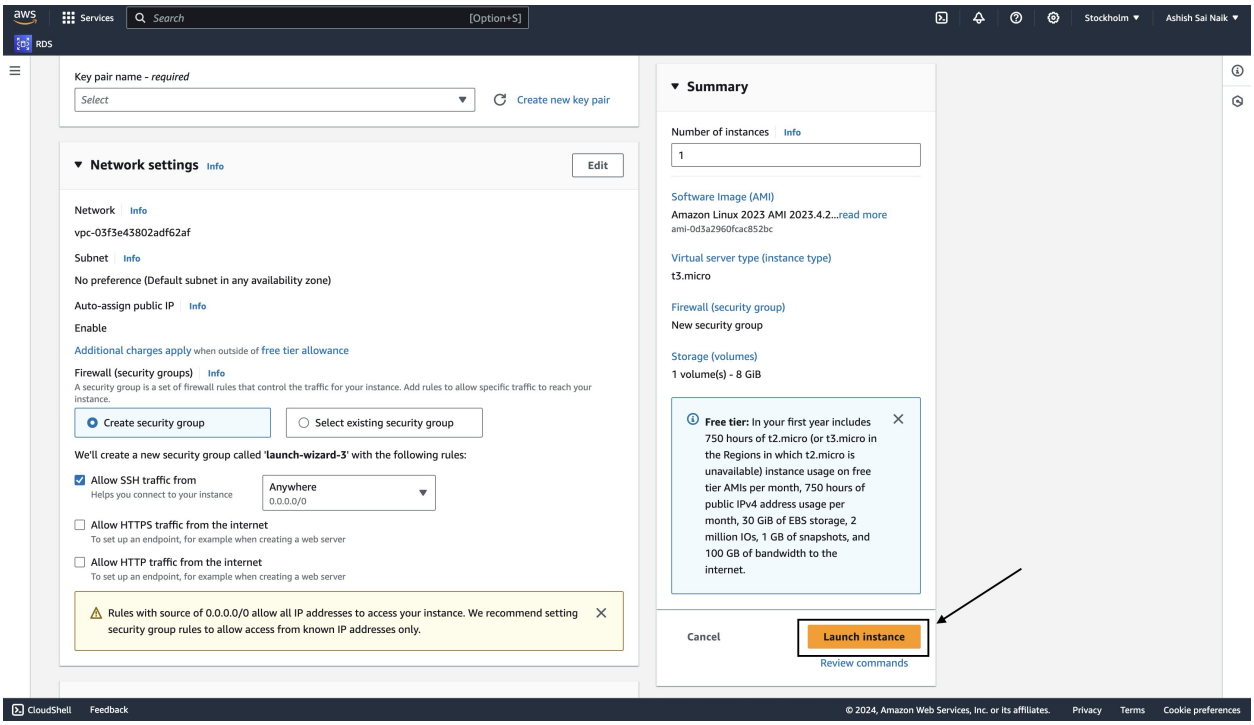
- Then in the Instance type you have to opt the t3.micro , its for free tire eligible

step-4

- Then create the Key pair

> ## Purpose
>
> A key pair consists of a public key and a private key. It is used to securely SSH (Secure Shell) into your EC2 instances without the need for a password. This enhances security and simplifies the login process.



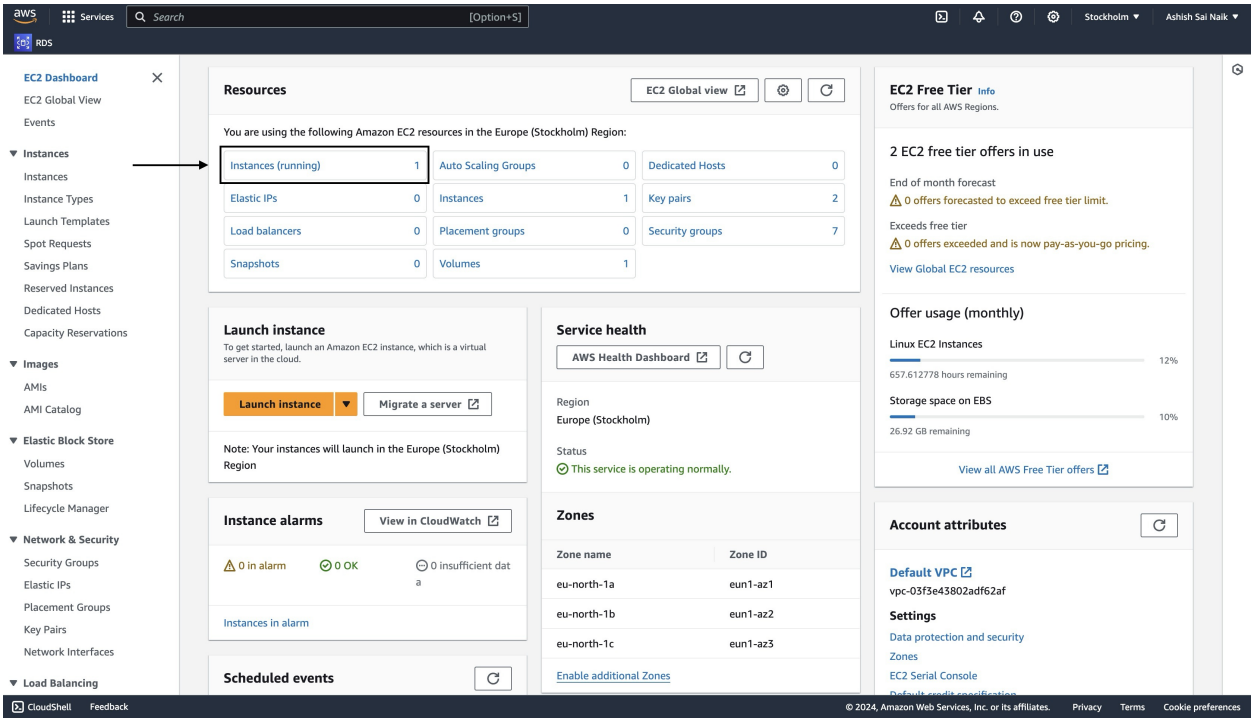


Then in the Network Settings make sure the enable Allow SHH traffic From “Anywhere” so we can access the website

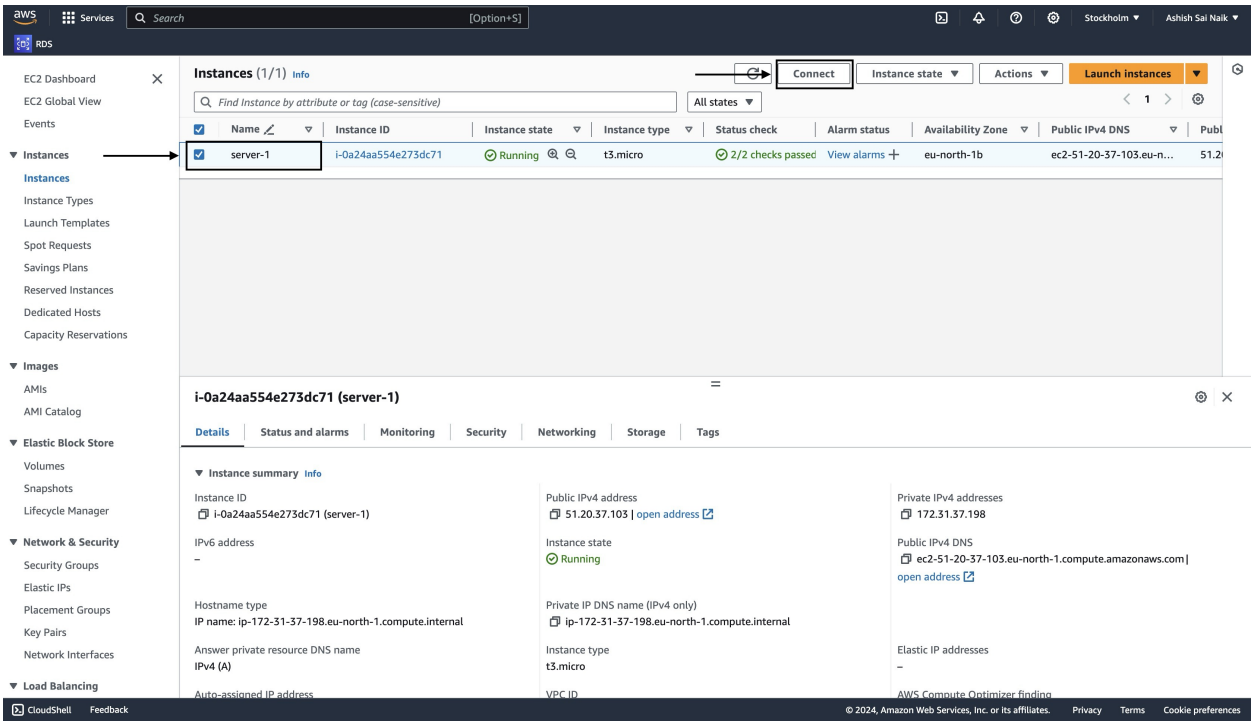
Finally we are ready to launch instance , which present in the right corner of the website

Navigate to Instances then check status of running instance

After successfully Launching the instance , return to the EC2 Dashboard Navigate to instance (running)

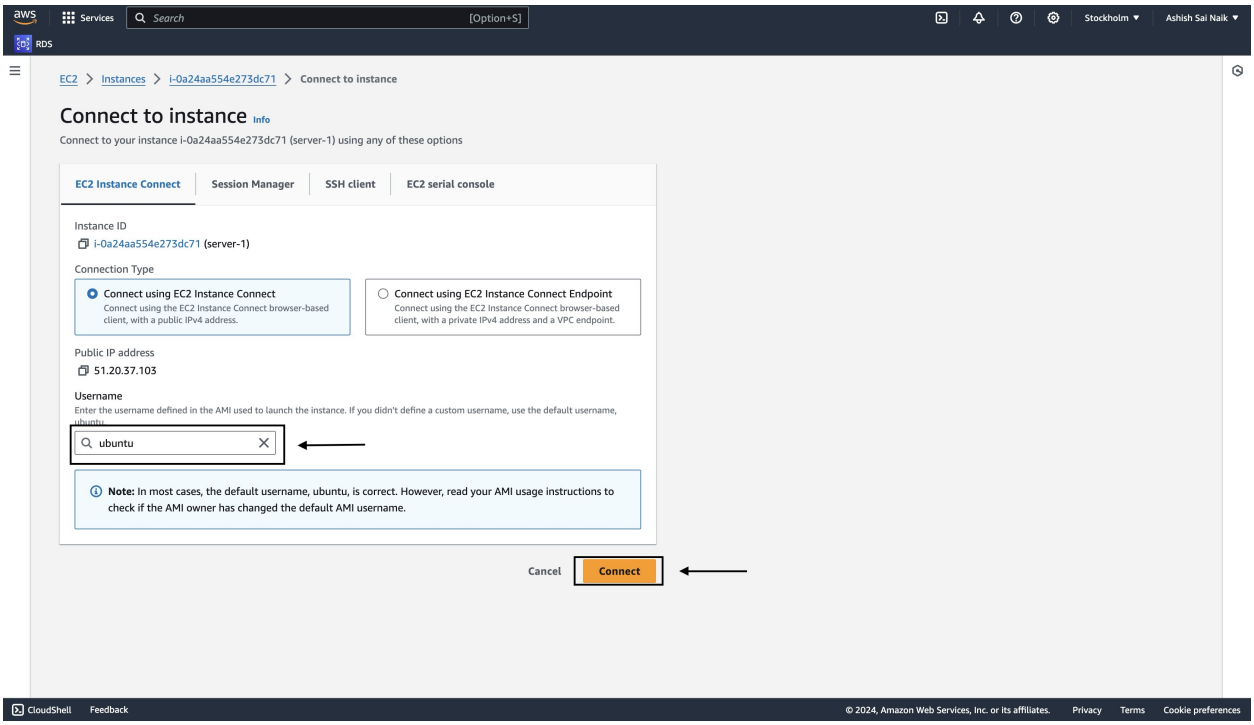


After redirecting into the Instance(running), Select the Instance which we want to connect to our Django project, Then Click the “connect“ .

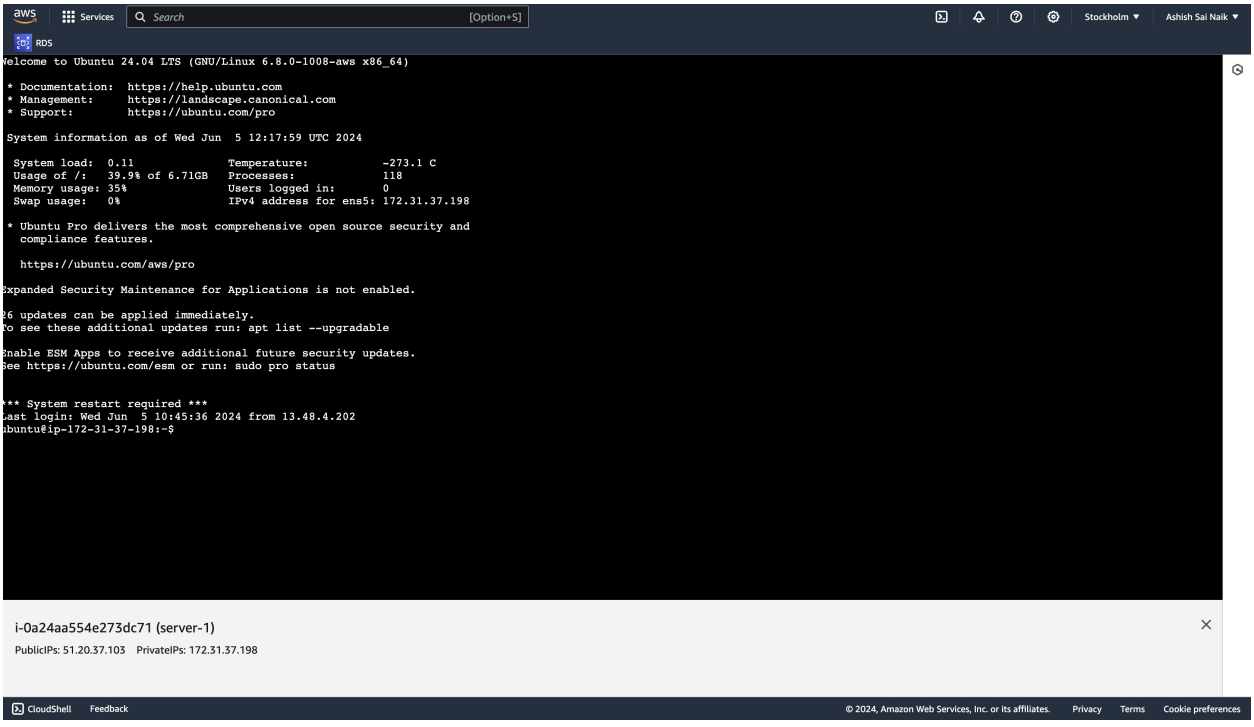


Once you clicked the connect , we will redirect to the below page .

Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ubuntu. then we are ready to connect.



After we connect, we will followed to the terminal



Install Packages

Since linux ami has its own python pre installed, I did not include it in my installations. I only added pip and django

```
sudo apt update
```

```
sudo apt-get install python3-pip
```

```
sudo pip3 install gunicorn
```

```
sudo apt-get install supervisor
```

```
sudo pip3 install django
```

```
sudo yum install nginx
```

Once we are done with the installing the packages , then we are ready to Clone the Django project in the git terminal

git clone -b Branch-1 https://github.com/yaswanthkoneri/job-portal-django.git

Download the django

```
sudo apt install python3-pip -y
```

```
pip install django
```

check, Does the git have been successful clone or not by

ls -lrt

this commada show the project name , alone with the total number of the files in it.

```
python3 manage.py makemigrations
```

```
python3 manage.py migrate
```

This create the all the migrations file required to run the app

One last step and our Job-portal-django project will be live by creating the Admin user to run this app. On the terminal, type the following command and provide username, email, password and once again password

```
python3 manage.py createsuperuser
```

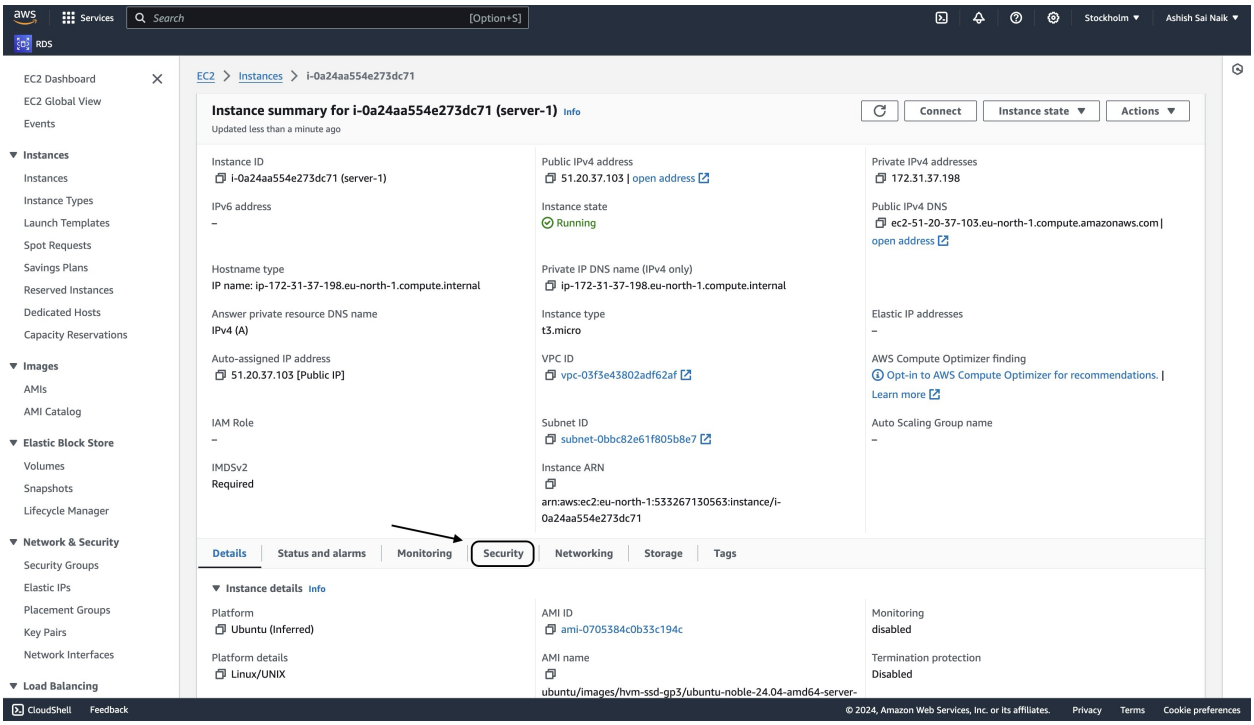
now its make app Live , we just need to start the server by the following the command line

```
python3 manage.py runserver
```

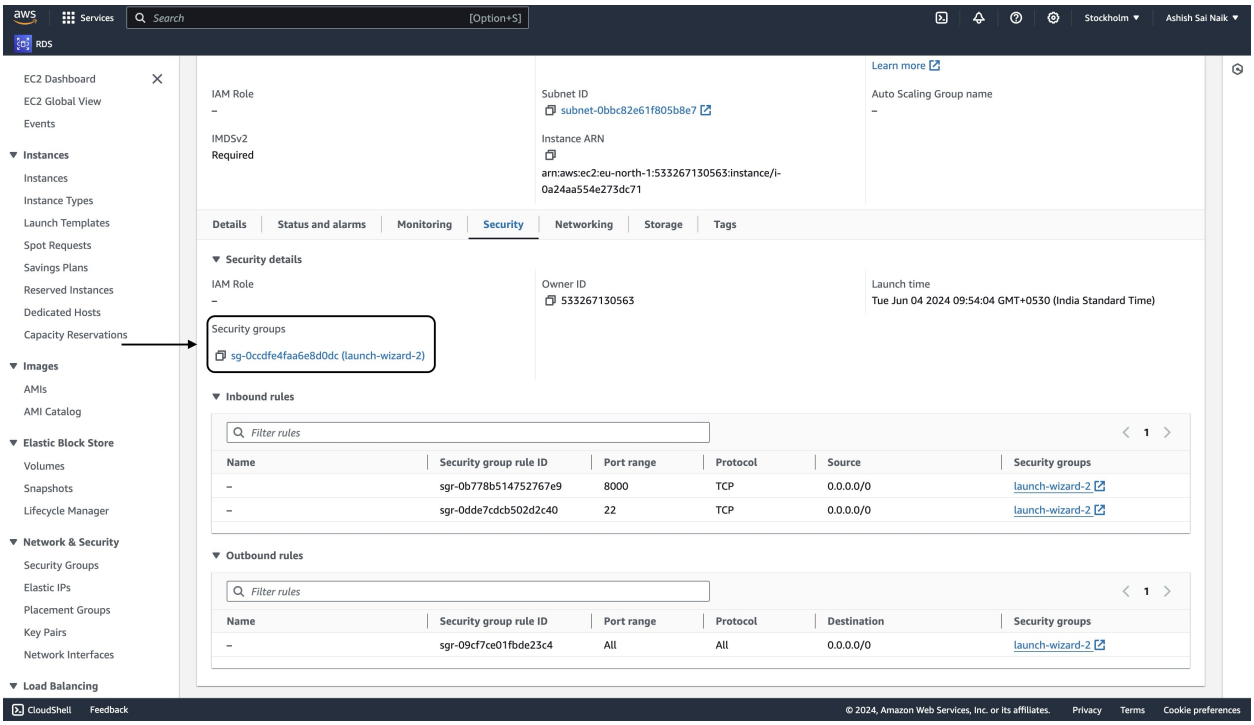
Once server hosted , we will head to the http://127.0.0.1:8000/

Since its still hosted in the local host so we have to change to the anywhere, we have to edit the security inbound rules, follow the step to change the inbound rule.

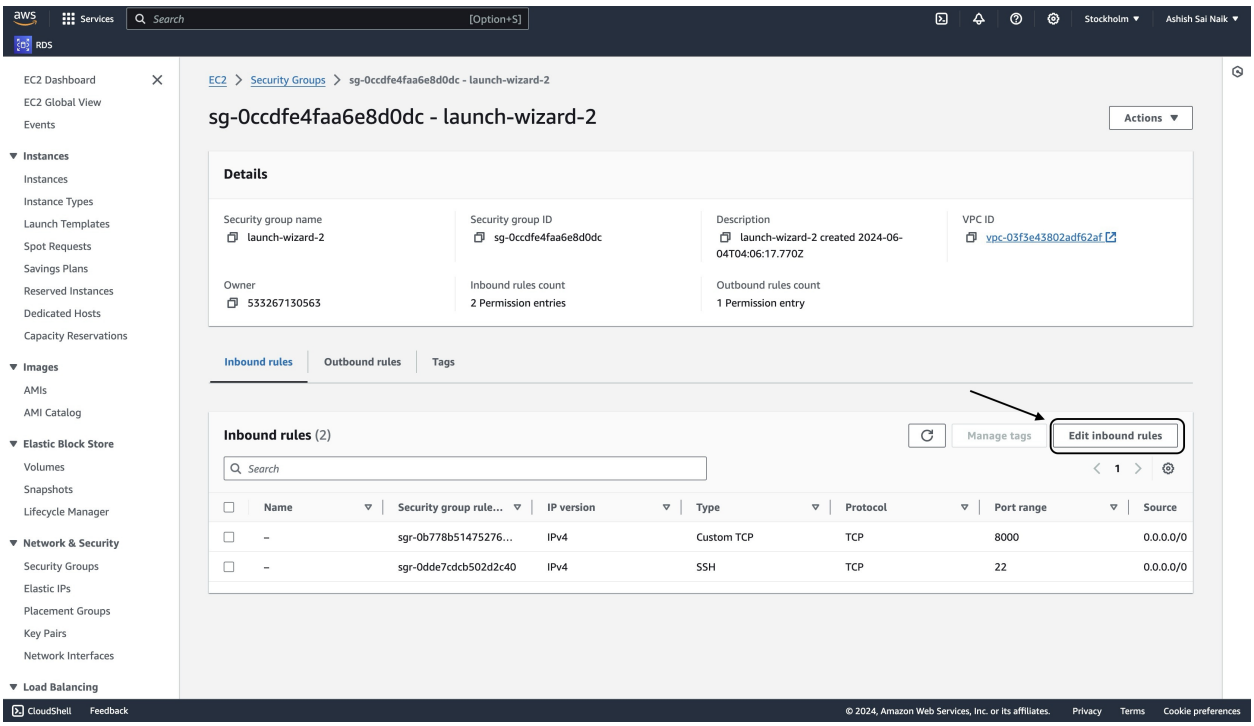
Step-1



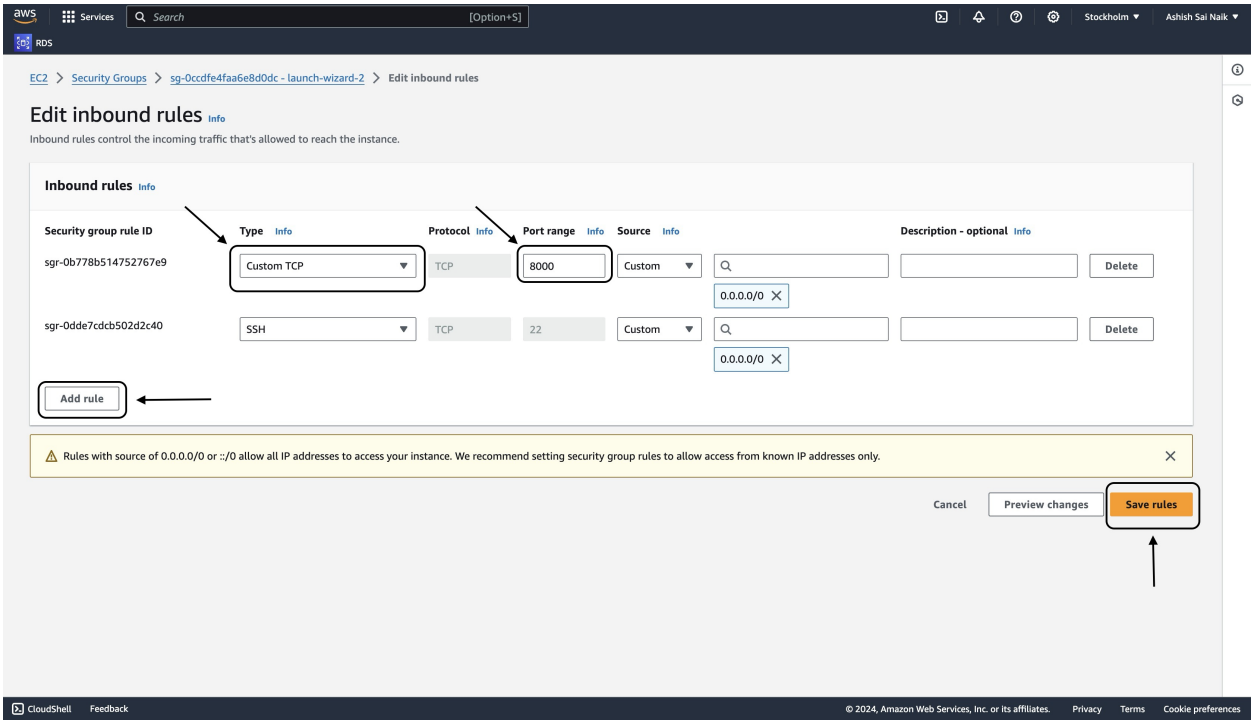
step-2


Services
Search
[Option+S]
Stockholm
Ashish Sai Naik
RDS
EC2 Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
IAM Role
–
IMDSv2
Required
Subnet ID
subnet-0bbc82e61f805b8e7
Instance ARN
arn:aws:ec2:eu-north-1:533267130563:instance/i-0a24aa554e273dc71
Learn more
Auto Scaling Group name
–
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Security details
IAM Role
–
Owner ID
533267130563
Launch time
Tue Jun 04 2024 09:54:04 GMT+0530 (India Standard Time)
Security groups
sg-0ccdfe4faa6e8d0dc (launch-wizard-2)
Inbound rules
Filter rules
Name
Security group rule ID
Port range
Protocol
Source
Security groups
– sgr-0b778b514752767e9 8000 TCP 0.0.0.0/0 launch-wizard-2
– sgr-0dde7cdcb502d2c40 22 TCP 0.0.0.0/0 launch-wizard-2
Outbound rules
Filter rules
Name
Security group rule ID
Port range
Protocol
Destination
Security groups
– sgr-09cf7ce01fbde23c4 All All 0.0.0.0/0 launch-wizard-2
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

step-3


Services
Search
[Option+S]
Stockholm
Ashish Sai Naik
RDS
EC2 Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
EC2 > Security Groups > sg-0ccdfe4faa6e8d0dc - launch-wizard-2
sg-0ccdfe4faa6e8d0dc - launch-wizard-2
Actions
Details
Security group name
launch-wizard-2
Security group ID
sg-0ccdfe4faa6e8d0dc
Description
launch-wizard-2 created 2024-06-04T04:06:17.770Z
VPC ID
vpc-03f3e43802adf62af
Owner
533267130563
Inbound rules count
2 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Tags
Inbound rules (2)
Manage tags
Edit inbound rules
Search
Name
Security group rule...
IP version
Type
Protocol
Port range
Source
– sgr-0b778b51475276... IPv4 Custom TCP TCP 8000 0.0.0.0/0
– sgr-0dde7cdcb502d2c40 IPv4 SSH TCP 22 0.0.0.0/0
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

step-4



Click on the Add rule ,Make sure in the Type select the “Custom TCP” and post range as 8000 and Then good to go with save rules.

After completing with changing inbound rule , we have to update the code in the code

from jobportal>>setting.py update the Allowed_Host to all the IP address,

```
ALLOWED_HOSTS = ['0.0.0.0','localhost',
    '127.0.0.1','51.20.37.103']
```

Then return to the terminal run the server by the following command line

`python3 manage.py runserver 0.0.0.0.8000`

Finally our project get live in the AWS Cloud

Page not found (404)

Directory indexes are not allowed here.

Request Method:	GET
Request URL:	http://51.20.37.103:8000/
Raised by:	django.views.static.serve

Using the URLconf defined in `jobportal.urls`, Django tried these URL patterns, in this order:

1. admin/
2. jobs/
3. ^(?P<path>.*)$

The empty path matched the last one.

You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.