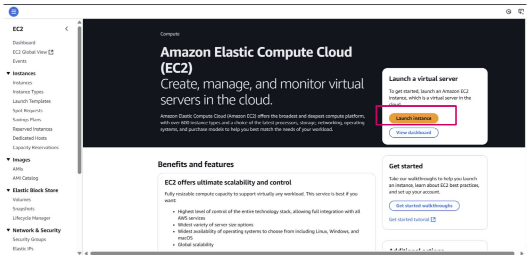
EC2- Hosting without Port

27 April 2025 20:15

Step 1: Go to EC2 Service



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

Compute

Amazon Elastic Compute Cloud (EC2)

Create, manage, and monitor virtual servers in the cloud.

Amazon Elastic Compute Cloud (Amazon EC2) offers the broadest and deepest compute platform, with over 600 instance types and a choice of the latest processors, storage, networking, operating systems, and purchase models to help you best match the needs of your workload.

Launch a virtual server

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Launch instance

View dashboard

Benefits and features

EC2 offers ultimate scalability and control

Fully resizable compute capacity to support virtually any workload. This service is best if you want:

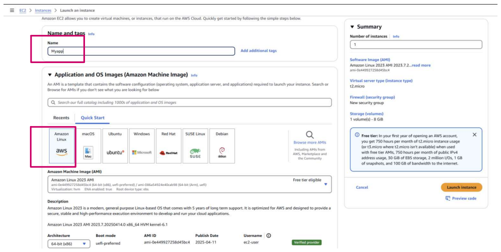
- Highest level of control of the entire technology stack, allowing full integration with all AWS services
- Widest variety of server size options
- Widest availability of operating systems to choose from including Linux, Windows, and macOS
- Global scalability

Get started

Take our walkthroughs to help you launch an instance, learn about EC2 best practices, and set up your account.

Get started walkthroughs

Get started tutorial



EC2 > Instances > Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: Myapp

Add additional tags

▼ Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents | Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Linux | Debian

Browse more AMIs — Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI — Free tier eligible
ami-0e449927258d45bc4 (64-bit (x86), uefi-preferred) / ami-086a54924e40cab98 (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.7.20250414.0 x86_64 HVM kernel-6.1

Architecture	Boot mode	AMI ID	Publish Date	Username	
64-bit (x86)	uefi-preferred	ami-0e449927258d45bc4	2025-04-11	ec2-user	Verified provider

▼ Summary

Number of instances: 1

Software Image (AMI): Amazon Linux 2023 AMI 2023.7.2...read more ami-0e449927258d45bc4

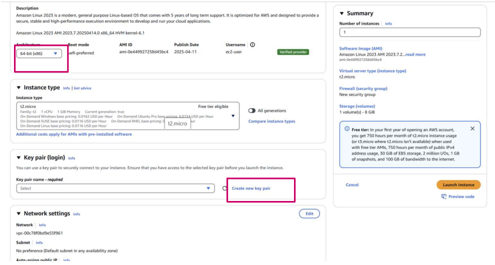
Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel | Launch instance | Preview code



Description

Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.7.20250414.0 x86_64 HVM kernel-6.1

Architecture	Boot mode	AMI ID	Publish Date	Username	
64-bit (x86)	uefi-preferred	ami-0e449927258d45bc4	2025-04-11	ec2-user	Verified provider

▼ Instance type Info | Get advice

Instance type

t2.micro — Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour On-Demand RHEL base pricing: 0 t2.micro ur
On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

Compare instance types

Additional costs apply for AMIs with pre-installed software

▼ Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*: Select

Create new key pair

▼ Network settings Info — Edit

Network: vpc-00c78f0bd9e55f961

Subnet: No preference (Default subnet in any availability zone)

Auto-assign public IP

▼ Summary

Number of instances: 1

Software Image (AMI): Amazon Linux 2023 AMI 2023.7.2...read more ami-0e449927258d45bc4

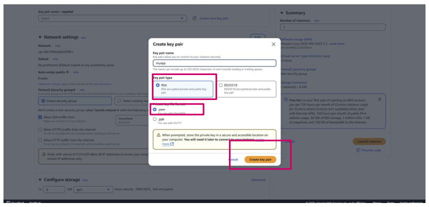
Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel | Launch instance | Preview code



Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.

myapp

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

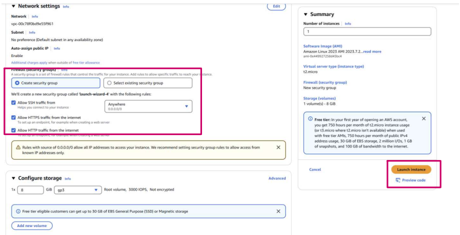
● RSA — RSA encrypted private and public key pair
○ ED25519 — ED25519 encrypted private and public key pair

Private key file format

● .pem — For use with OpenSSH
○ .ppk — For use with PuTTY

When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** Learn more

Cancel | Create key pair



▼ Network settings Info — Edit

Network: vpc-00c78f0bd9e55f961

Subnet: No preference (Default subnet in any availability zone)

Auto-assign public IP: Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

● Create security group ○ Select existing security group

We'll create a new security group called '**launch-wizard-4**' with the following rules:

☑ Allow SSH traffic from — Helps you connect to your instance — Anywhere 0.0.0.0/0

☑ Allow HTTPS traffic from the internet — To set up an endpoint, for example when creating a web server

☑ Allow HTTP traffic from the internet — To set up an endpoint, for example when creating a web server

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

▼ Configure storage Info — Advanced

1x 8 GiB gp3 Root volume, 3000 IOPS, Not encrypted

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

Add new volume

▼ Summary

Number of instances: 1

Software Image (AMI): Amazon Linux 2023 AMI 2023.7.2...read more ami-0e449927258d45bc4

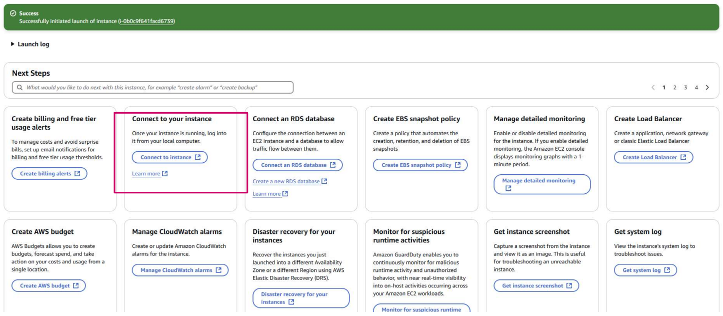
Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel | Launch instance | Preview code



Success
Successfully initiated launch of instance (i-0b0c9f641facd6739)

▶ Launch log

Next Steps

What would you like to do next with this instance, for example "create alarm" or "create backup"

< 1 2 3 4 >

Create billing and free tier usage alerts — To manage costs and avoid surprise bills, set up email notifications for billing and free tier usage thresholds. — Create billing alerts

Connect to your instance — Once your instance is running, log into it from your local computer. — Connect to instance — Learn more

Connect an RDS database — Configure the connection between an EC2 instance and a database to allow traffic flow between them. — Connect an RDS database — Create a new RDS database — Learn more

Create EBS snapshot policy — Create a policy that automates the creation, retention, and deletion of EBS snapshots — Create EBS snapshot policy

Manage detailed monitoring — Enable or disable detailed monitoring for the instance. If you enable detailed monitoring, the Amazon EC2 console displays monitoring graphs with a 1-minute period. — Manage detailed monitoring

Create Load Balancer — Create a application, network gateway or classic Elastic Load Balancer — Create Load Balancer

Create AWS budget — AWS Budgets allows you to create budgets, forecast spend, and take action on your costs and usage from a single location. — Create AWS budget

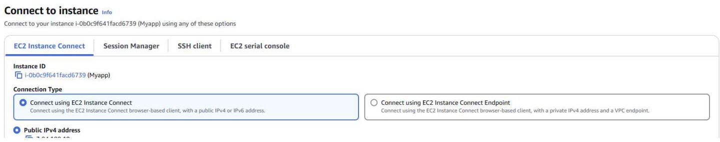
Manage CloudWatch alarms — Create or update Amazon CloudWatch alarms for the instance. — Manage CloudWatch alarms

Disaster recovery for your instances — Recover the instances you just launched into a different Availability Zone or a different Region using AWS Elastic Disaster Recovery (DRS). — Disaster recovery for your instances

Monitor for suspicious runtime activities — Amazon GuardDuty enables you to continuously monitor for malicious runtime activity and unauthorized behavior, with near real-time visibility into on-host activities occurring across your Amazon EC2 workloads. — Monitor for suspicious runtime

Get instance screenshot — Capture a screenshot from the instance and view it as an image. This is useful for troubleshooting an unreachable instance. — Get instance screenshot

Get system log — View the instance's system log to troubleshoot issues. — Get system log



Connect to instance Info

Connect to your instance i-0b0c9f641facd6739 (Myapp) using any of these options

EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

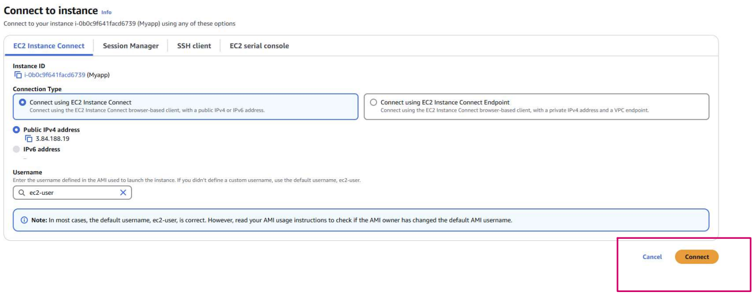
Instance ID: i-0b0c9f641facd6739 (Myapp)

Connection Type

● Connect using EC2 Instance Connect — Connect using the EC2 Instance Connect browser-based client, with a public IPv4 or IPv6 address.

○ Connect using EC2 Instance Connect Endpoint — Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

● Public IPv4 address



Connect to instance

Connect to your instance i-0b0c9f641facd6739 (Myapp) using any of these options

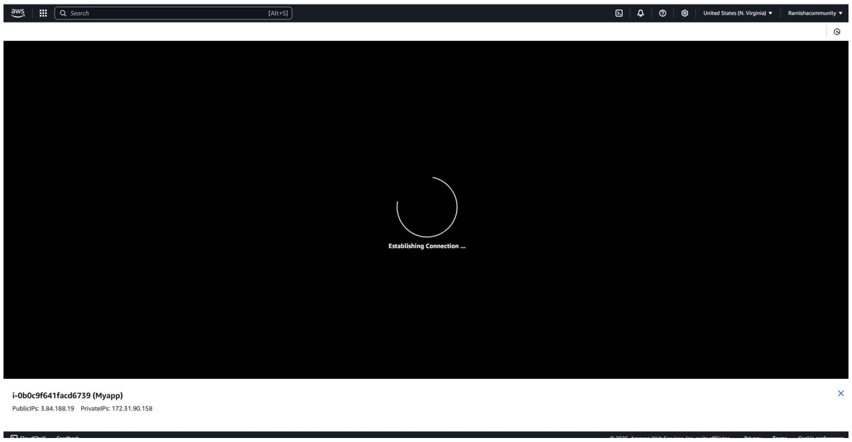
Instance ID
i-0b0c9f641facd6739 (Myapp)

Connection Type
● Connect using EC2 Instance Connect
Connect using the EC2 Instance Connect browser-based client, with a public IPv4 or IPv6 address.
○ Connect using EC2 Instance Connect Endpoint
Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

● Public IPv4 address
3.84.188.19
○ IPv6 address

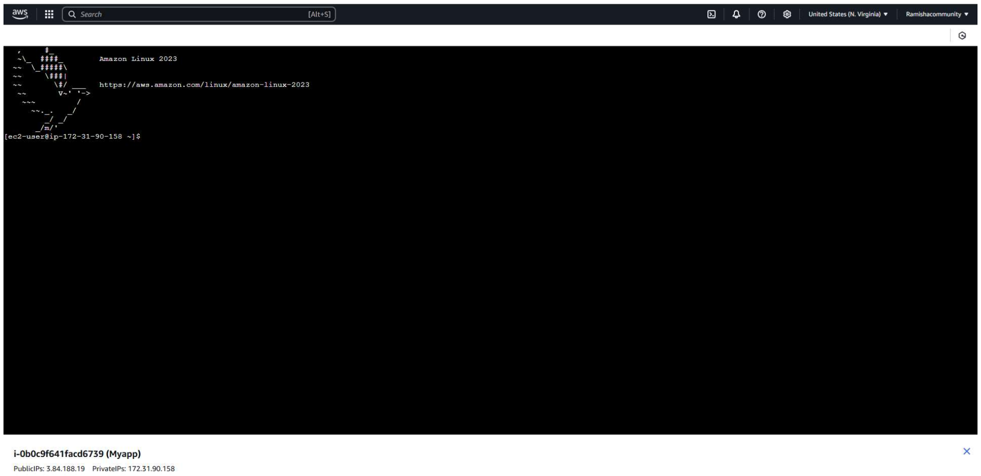
Username
Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ec2-user.
ec2-user

Note: In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.



Establishing Connection ...

i-0b0c9f641facd6739 (Myapp)
PublicIPs: 3.84.188.19 PrivateIPs: 172.31.90.158

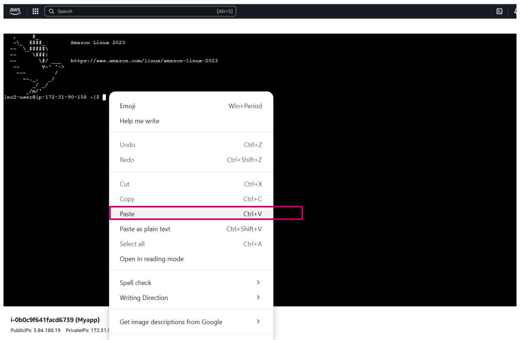


```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-90-158 ~]$
```

i-0b0c9f641facd6739 (Myapp)
PublicIPs: 3.84.188.19 PrivateIPs: 172.31.90.158

```
sudo yum update -y
```

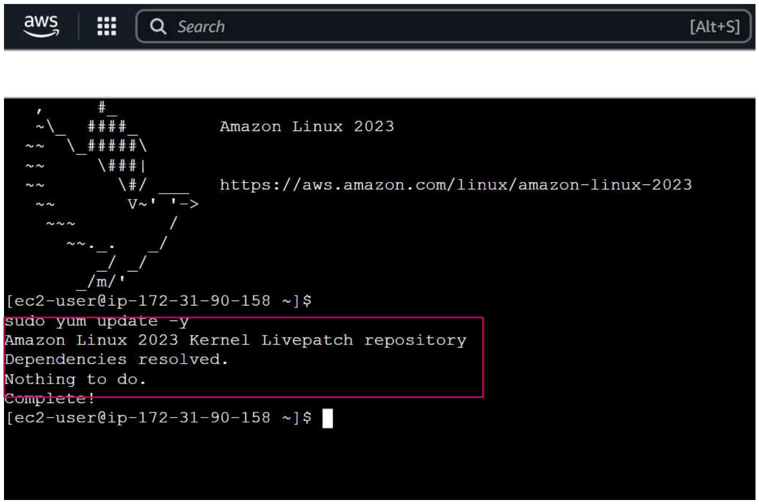
Copy the command and Right Click it and paste it



```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-90-158 ~]$
```

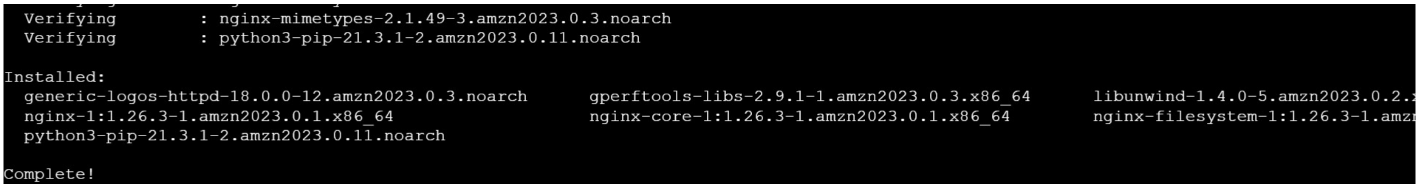
i-0b0c9f641facd6739 (Myapp)
PublicIPs: 3.84.188.19 PrivateIPs: 172.31.

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
```



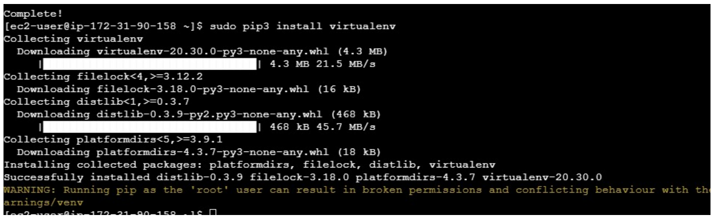
```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-90-158 ~]$
sudo yum update -y
Amazon Linux 2023 Kernel Livepatch repository
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-90-158 ~]$
```

sudo yum install python3 python3-pip nginx -y



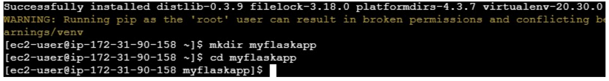
```
  Verifying        : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch
  Verifying        : python3-pip-21.3.1-2.amzn2023.0.11.noarch

Installed:
  generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch      gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64      libunwind-1.4.0-5.amzn2023.0.2.:
  nginx-1:1.26.3-1.amzn2023.0.1.x86_64                   nginx-core-1:1.26.3-1.amzn2023.0.1.x86_64        nginx-filesystem-1:1.26.3-1.amzr
  python3-pip-21.3.1-2.amzn2023.0.11.noarch

Complete!
```

sudo pip3 install virtualenv



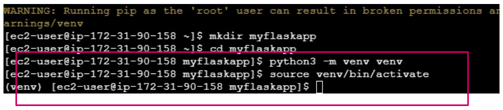
```
Complete!
[ec2-user@ip-172-31-90-158 ~]$ sudo pip3 install virtualenv
Collecting virtualenv
  Downloading virtualenv-20.30.0-py3-none-any.whl (4.3 MB)
     |████████████████████████████████| 4.3 MB 21.5 MB/s
Collecting filelock<4,>=3.12.2
  Downloading filelock-3.18.0-py3-none-any.whl (16 kB)
Collecting distlib<1,>=0.3.7
  Downloading distlib-0.3.9-py2.py3-none-any.whl (468 kB)
     |████████████████████████████████| 468 kB 45.7 MB/s
Collecting platformdirs<5,>=3.9.1
  Downloading platformdirs-4.3.7-py3-none-any.whl (18 kB)
Installing collected packages: platformdirs, filelock, distlib, virtualenv
Successfully installed distlib-0.3.9 filelock-3.18.0 platformdirs-4.3.7 virtualenv-20.30.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the
arnings/venv
[ec2-user@ip-172-31-90-158 ~]$
```

mkdir myflaskapp
cd myflaskapp



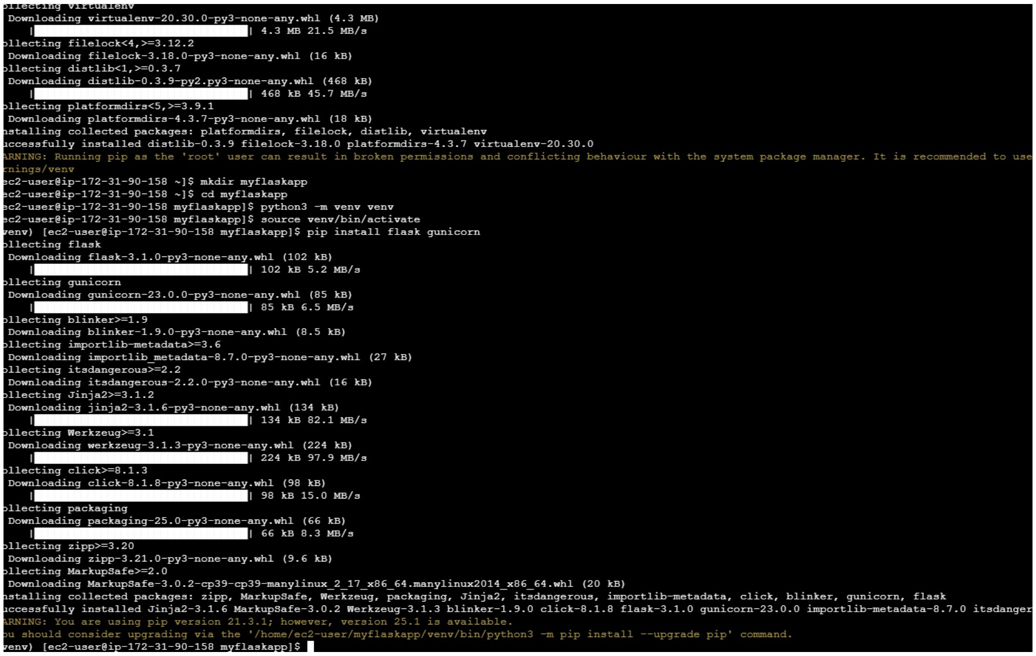
```
Successfully installed distlib-0.3.9 filelock-3.18.0 platformdirs-4.3.7 virtualenv-20.30.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting be
arnings/venv
[ec2-user@ip-172-31-90-158 ~]$ mkdir myflaskapp
[ec2-user@ip-172-31-90-158 ~]$ cd myflaskapp
[ec2-user@ip-172-31-90-158 myflaskapp]$
```

python3 -m venv venv
source venv/bin/activate



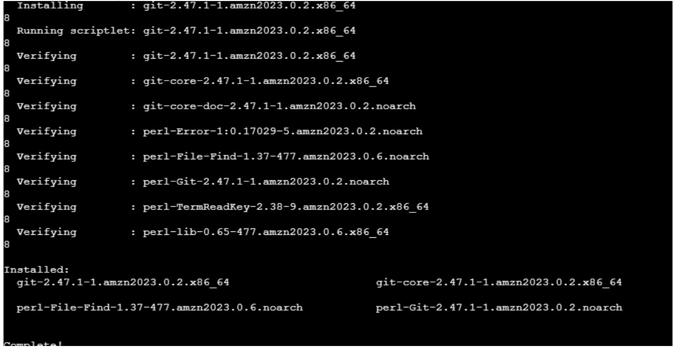
```
WARNING: Running pip as the 'root' user can result in broken permissions ar
arnings/venv
[ec2-user@ip-172-31-90-158 ~]$ mkdir myflaskapp
[ec2-user@ip-172-31-90-158 ~]$ cd myflaskapp
[ec2-user@ip-172-31-90-158 myflaskapp]$ python3 -m venv venv
[ec2-user@ip-172-31-90-158 myflaskapp]$ source venv/bin/activate
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$
```

pip install flask gunicorn



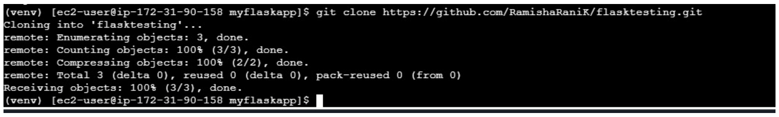
```
ollecting virtualenv
 Downloading virtualenv-20.30.0-py3-none-any.whl (4.3 MB)
     |████████████████████████████████| 4.3 MB 21.5 MB/s
ollecting filelock<4,>=3.12.2
 Downloading filelock-3.18.0-py3-none-any.whl (16 kB)
ollecting distlib<1,>=0.3.7
 Downloading distlib-0.3.9-py2.py3-none-any.whl (468 kB)
     |████████████████████████████████| 468 kB 45.7 MB/s
ollecting platformdirs<5,>=3.9.1
 Downloading platformdirs-4.3.7-py3-none-any.whl (18 kB)
nstalling collected packages: platformdirs, filelock, distlib, virtualenv
uccessfully installed distlib-0.3.9 filelock-3.18.0 platformdirs-4.3.7 virtualenv-20.30.0
ARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use
rnings/venv
ec2-user@ip-172-31-90-158 ~]$ mkdir myflaskapp
ec2-user@ip-172-31-90-158 ~]$ cd myflaskapp
ec2-user@ip-172-31-90-158 myflaskapp]$ python3 -m venv venv
ec2-user@ip-172-31-90-158 myflaskapp]$ source venv/bin/activate
venv) [ec2-user@ip-172-31-90-158 myflaskapp]$ pip install flask gunicorn
ollecting flask
 Downloading flask-3.1.0-py3-none-any.whl (102 kB)
     |████████████████████████████████| 102 kB 5.2 MB/s
ollecting gunicorn
 Downloading gunicorn-23.0.0-py3-none-any.whl (85 kB)
     |████████████████████████████████| 85 kB 6.5 MB/s
ollecting blinker>=1.9
 Downloading blinker-1.9.0-py3-none-any.whl (8.5 kB)
ollecting importlib-metadata>=3.6
 Downloading importlib_metadata-8.7.0-py3-none-any.whl (27 kB)
ollecting itsdangerous>=2.2
 Downloading itsdangerous-2.2.0-py3-none-any.whl (16 kB)
ollecting Jinja2>=3.1.2
 Downloading jinja2-3.1.6-py3-none-any.whl (134 kB)
     |████████████████████████████████| 134 kB 82.1 MB/s
ollecting Werkzeug>=3.1
 Downloading werkzeug-3.1.3-py3-none-any.whl (224 kB)
     |████████████████████████████████| 224 kB 97.9 MB/s
ollecting click>=8.1.3
 Downloading click-8.1.8-py3-none-any.whl (98 kB)
     |████████████████████████████████| 98 kB 15.0 MB/s
ollecting packaging
 Downloading packaging-25.0-py3-none-any.whl (66 kB)
     |████████████████████████████████| 66 kB 8.3 MB/s
ollecting zipp>=3.20
 Downloading zipp-3.21.0-py3-none-any.whl (9.6 kB)
ollecting MarkupSafe>=2.0
 Downloading MarkupSafe-3.0.2-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (20 kB)
nstalling collected packages: zipp, MarkupSafe, Werkzeug, packaging, Jinja2, itsdangerous, importlib-metadata, click, blinker, gunicorn, flask
uccessfully installed Jinja2-3.1.6 MarkupSafe-3.0.2 Werkzeug-3.1.3 blinker-1.9.0 click-8.1.8 flask-3.1.0 gunicorn-23.0.0 importlib-metadata-8.7.0 itsdanger
ARNING: You are using pip version 21.3.1; however, version 25.1 is available.
ou should consider upgrading via the '/home/ec2-user/myflaskapp/venv/bin/python3 -m pip install --upgrade pip' command.
venv) [ec2-user@ip-172-31-90-158 myflaskapp]$
```

sudo yum install git -y



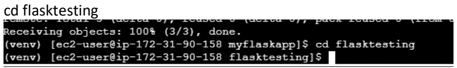
```
  Installing       : git-2.47.1-1.amzn2023.0.2.x86_64
8
  Running scriptlet: git-2.47.1-1.amzn2023.0.2.x86_64
8
  Verifying        : git-2.47.1-1.amzn2023.0.2.x86_64
8
  Verifying        : git-core-2.47.1-1.amzn2023.0.2.x86_64
8
  Verifying        : git-core-doc-2.47.1-1.amzn2023.0.2.noarch
8
  Verifying        : perl-Error-1:0.17029-5.amzn2023.0.2.noarch
8
  Verifying        : perl-File-Find-1.37-477.amzn2023.0.6.noarch
8
  Verifying        : perl-Git-2.47.1-1.amzn2023.0.2.noarch
8
  Verifying        : perl-TermReadKey-2.38-9.amzn2023.0.2.x86_64
8
  Verifying        : perl-lib-0.65-477.amzn2023.0.6.x86_64
8

Installed:
  git-2.47.1-1.amzn2023.0.2.x86_64                      git-core-2.47.1-1.amzn2023.0.2.x86_64

  perl-File-Find-1.37-477.amzn2023.0.6.noarch           perl-Git-2.47.1-1.amzn2023.0.2.noarch


Complete!
```

git clone https://github.com/RamishaRaniK/flasktesting.git
git clone https://github.com/RamishaRaniK/awslocalmodelhosting.git



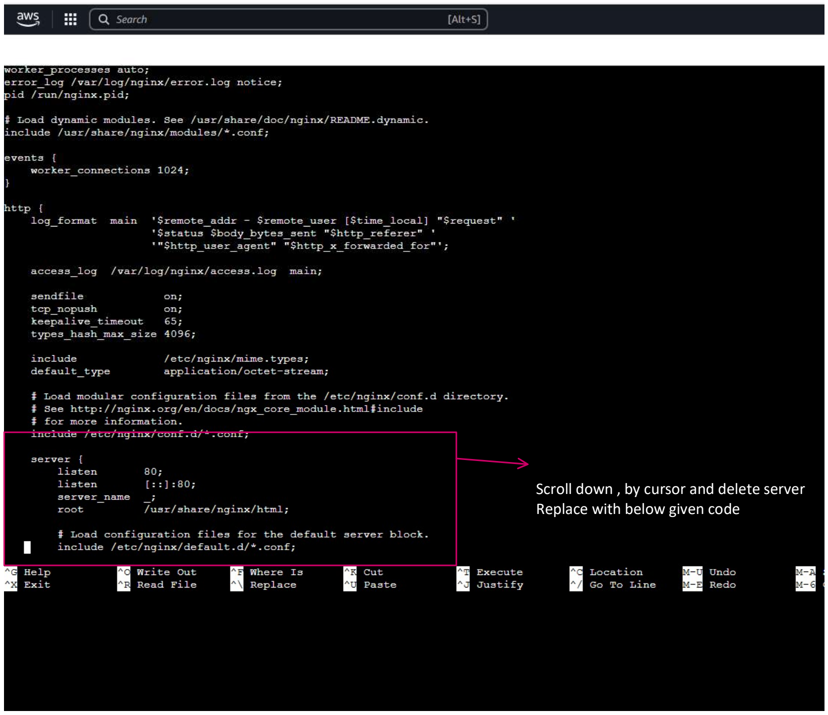
```
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$ git clone https://github.com/RamishaRaniK/flasktesting.git
Cloning into 'flasktesting'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$
```

cd flasktesting



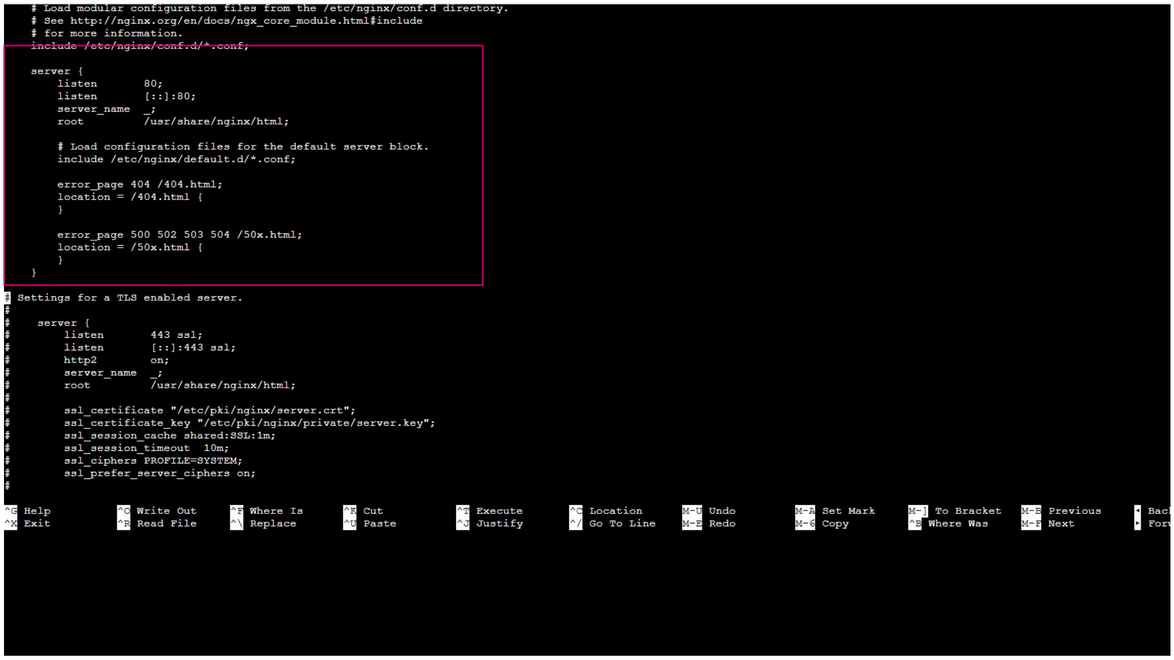
```
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0
Receiving objects: 100% (3/3), done.
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$ cd flasktesting
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$
```

pip install -r requirements.txt
sudo nano /etc/nginx/nginx.conf



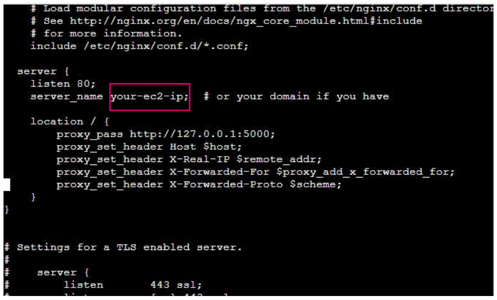
```
worker_processes auto;
error_log /var/log/nginx/error.log notice;
pid /run/nginx.pid;

# Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.
include /usr/share/nginx/modules/*.conf;

events {
    worker_connections 1024;
}

http {
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;

    sendfile            on;
    tcp_nopush          on;
    keepalive_timeout   65;
    types_hash_max_size 4096;

    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;

    server {
        listen       80;
        listen       [::]:80;
        server_name  _;
        root         /usr/share/nginx/html;

        # Load configuration files for the default server block.
        include /etc/nginx/default.d/*.conf;
```

Scroll down , by cursor and delete server
Replace with below given code

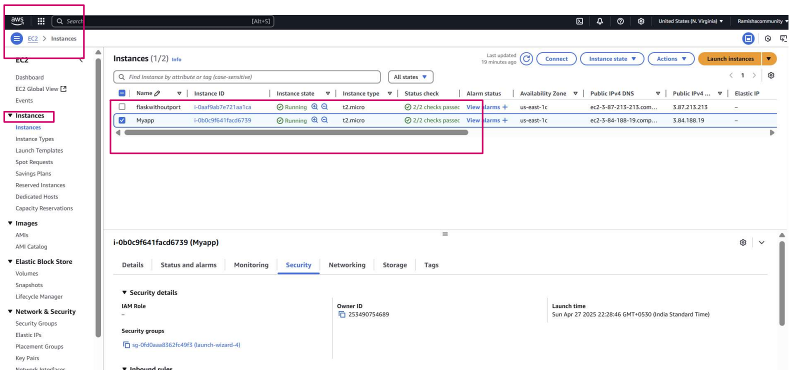


```
    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;

    server {
        listen       80;
        listen       [::]:80;
        server_name  _;
        root         /usr/share/nginx/html;

        # Load configuration files for the default server block.
        include /etc/nginx/default.d/*.conf;

        error_page 404 /404.html;
        location = /404.html {
        }

        error_page 500 502 503 504 /50x.html;
        location = /50x.html {
        }
    }

# Settings for a TLS enabled server.
#
#    server {
#        listen       443 ssl;
#        listen       [::]:443 ssl;
#        http2        on;
#        server_name  _;
#        root         /usr/share/nginx/html;
#
#        ssl_certificate "/etc/pki/nginx/server.crt";
#        ssl_certificate_key "/etc/pki/nginx/private/server.key";
#        ssl_session_cache shared:SSL:1m;
#        ssl_session_timeout  10m;
#        ssl_ciphers PROFILE=SYSTEM;
#        ssl_prefer_server_ciphers on;
#
```

```
server {
    listen 80;
    server_name your-ec2-ip;  # or your domain if you have

    location / {
        proxy_pass http://127.0.0.1:5000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```



```
    # Load modular configuration files from the /etc/nginx/conf.d director
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;

  server {
    listen 80;
    server_name your-ec2-ip;  # or your domain if you have

    location / {
        proxy_pass http://127.0.0.1:5000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}

# Settings for a TLS enabled server.
#
#    server {
#        listen       443 ssl;
```



Instances (1/2) Info

Last updated 19 minutes ago

Find Instance by attribute or tag (case-sensitive)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
flaskwithoutport	i-0aaf9ab7e721aa1ca	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c	ec2-3-87-213-213.com...	3.87.213.213	–
Myapp	i-0b0c9f641facd6739	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c	ec2-3-84-188-19.comp...	3.84.188.19	–

i-0b0c9f641facd6739 (Myapp)

Security details

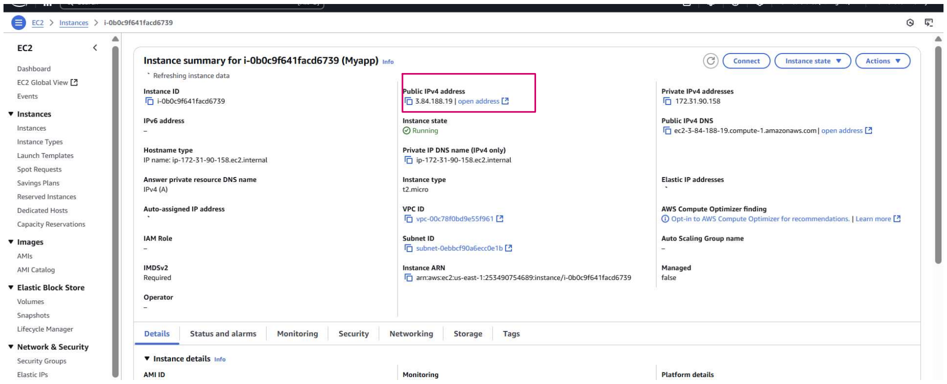
IAM Role
–

Owner ID
253490754689

Launch time
Sun Apr 27 2025 22:28:46 GMT+0530 (India Standard Time)

Security groups
sg-0fd0aaa8362fc49f3 (launch-wizard-4)

Inbound rules



Instance summary for i-0b0c9f641facd6739 (Myapp) Info

Refreshing instance data

Instance ID
i-0b0c9f641facd6739

Public IPv4 address
3.84.188.19 | open address

Private IPv4 addresses
172.31.90.158

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-3-84-188-19.compute-1.amazonaws.com | open address

Hostname type
IP name: ip-172-31-90-158.ec2.internal

Private IP DNS name (IPv4 only)
ip-172-31-90-158.ec2.internal

Answer private resource DNS name
IPv4 (A)

Instance type
t2.micro

Elastic IP addresses

Auto-assigned IP address

VPC ID
vpc-00c78f0bd9e55f961

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

IAM Role
–

Subnet ID
subnet-0ebbcf90a6ecc0e1b

Auto Scaling Group name
–

IMDSv2
Required

Instance ARN
arn:aws:ec2:us-east-1:253490754689:instance/i-0b0c9f641facd6739

Managed
false

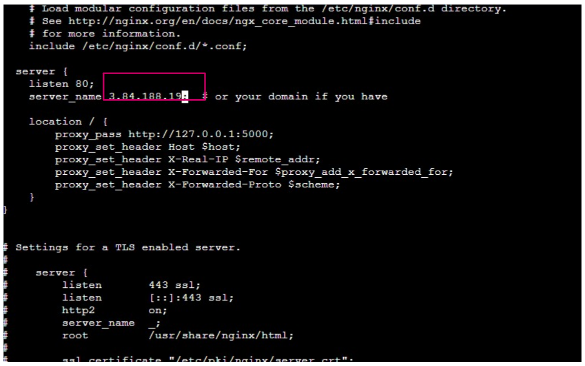
Operator
–

Instance details Info

AMI ID

Monitoring

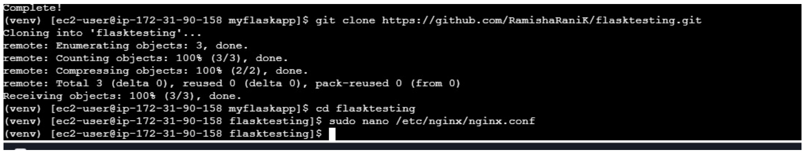
Platform details



```
    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;

  server {
    listen 80;
    server_name 3.84.188.19;  # or your domain if you have

    location / {
        proxy_pass http://127.0.0.1:5000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}

# Settings for a TLS enabled server.
#
#    server {
#        listen       443 ssl;
#        listen       [::]:443 ssl;
#        http2        on;
#        server_name  _;
#        root         /usr/share/nginx/html;
#
#        ssl_certificate "/etc/pki/nginx/server.crt";
```

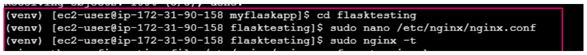
Ctrl+S

Ctrl+X --? exit



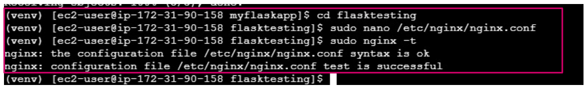
```
Complete!
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$ git clone https://github.com/RamishaRaniK/flasktesting.git
Cloning into 'flasktesting'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$ cd flasktesting
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ sudo nano /etc/nginx/nginx.conf
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$
```

sudo nginx -t



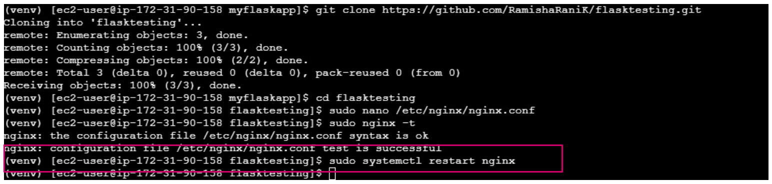
```
Receiving objects: 100% (3/3), done.
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$ cd flasktesting
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ sudo nano /etc/nginx/nginx.conf
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ sudo nginx -t
```

sudo nginx -t



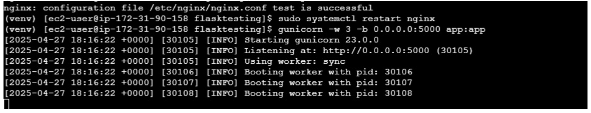
```
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$ cd flasktesting
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ sudo nano /etc/nginx/nginx.conf
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$
```

sudo systemctl restart nginx

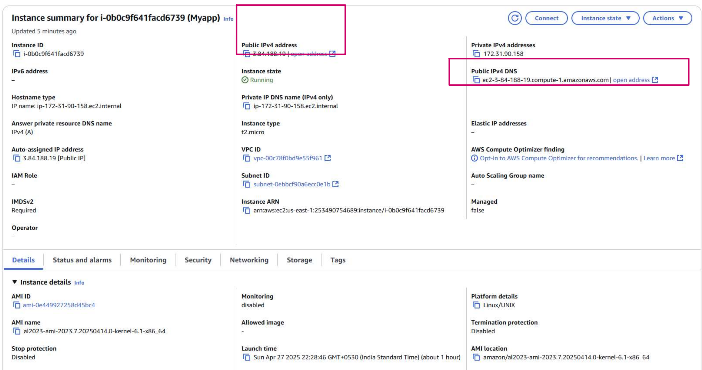


```
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$ git clone https://github.com/RamishaRaniK/flasktesting.git
Cloning into 'flasktesting'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
(venv) [ec2-user@ip-172-31-90-158 myflaskapp]$ cd flasktesting
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ sudo nano /etc/nginx/nginx.conf
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ sudo systemctl restart nginx
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$
```

gunicorn -w 3 -b 0.0.0.0:5000 app:app



```
nginx: configuration file /etc/nginx/nginx.conf test is successful
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ sudo systemctl restart nginx
(venv) [ec2-user@ip-172-31-90-158 flasktesting]$ gunicorn -w 3 -b 0.0.0.0:5000 app:app
[2025-04-27 18:16:22 +0000] [30105] [INFO] Starting gunicorn 23.0.0
[2025-04-27 18:16:22 +0000] [30105] [INFO] Listening at: http://0.0.0.0:5000 (30105)
[2025-04-27 18:16:22 +0000] [30105] [INFO] Using worker: sync
[2025-04-27 18:16:22 +0000] [30106] [INFO] Booting worker with pid: 30106
[2025-04-27 18:16:22 +0000] [30107] [INFO] Booting worker with pid: 30107
[2025-04-27 18:16:22 +0000] [30108] [INFO] Booting worker with pid: 30108
```



Instance summary for i-0b0c9f641facd6739 (Myapp) Info

Updated 5 minutes ago

Instance ID
i-0b0c9f641facd6739

Public IPv4 address
~~3.84.188.19 | open address~~

Private IPv4 addresses
172.31.90.158

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-3-84-188-19.compute-1.amazonaws.com | open address

Hostname type
IP name: ip-172-31-90-158.ec2.internal

Private IP DNS name (IPv4 only)
ip-172-31-90-158.ec2.internal

Answer private resource DNS name
IPv4 (A)

Instance type
t2.micro

Elastic IP addresses
–

Auto-assigned IP address
3.84.188.19 [Public IP]

VPC ID
vpc-00c78f0bd9e55f961

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

IAM Role
–

Subnet ID
subnet-0ebbcf90a6ecc0e1b

Auto Scaling Group name
–

IMDSv2
Required

Instance ARN
arn:aws:ec2:us-east-1:253490754689:instance/i-0b0c9f641facd6739

Managed
false

Operator
–

▼ Instance details Info

AMI ID
ami-0e449927258d45bc4

Monitoring
disabled

Platform details
Linux/UNIX

AMI name
al2023-ami-2023.7.20250414.0-kernel-6.1-x86_64

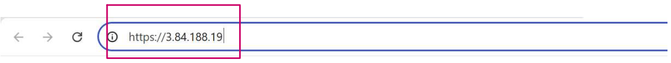
Allowed image
-

Termination protection
Disabled

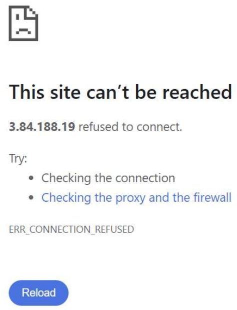
Stop protection
Disabled

Launch time
Sun Apr 27 2025 22:28:46 GMT+0530 (India Standard Time) (about 1 hour)

AMI location
amazon/al2023-ami-2023.7.20250414.0-kernel-6.1-x86_64



https://3.84.188.19



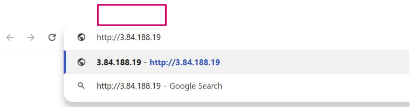
This site can’t be reached

3.84.188.19 refused to connect.

Try:

- Checking the connection
- Checking the proxy and the firewall

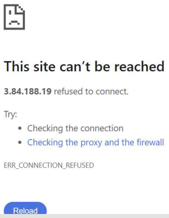
ERR_CONNECTION_REFUSED



http://3.84.188.19

3.84.188.19 - **http://3.84.188.19**

http://3.84.188.19 - Google Search



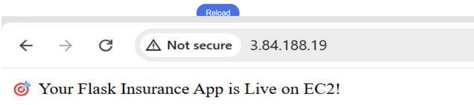
This site can’t be reached

3.84.188.19 refused to connect.

Try:

- Checking the connection
- Checking the proxy and the firewall

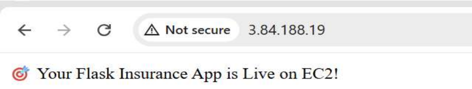
ERR_CONNECTION_REFUSED



Not secure 3.84.188.19

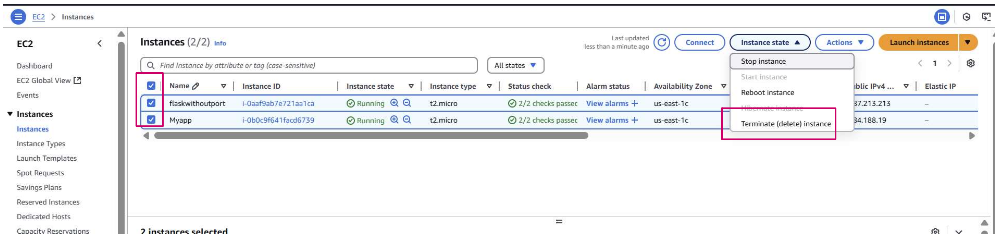
🎯 Your Flask Insurance App is Live on EC2!

If you are practicing delete it

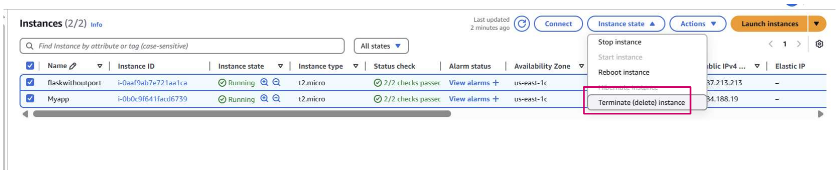


🎯 Your Flask Insurance App is Live on EC2!

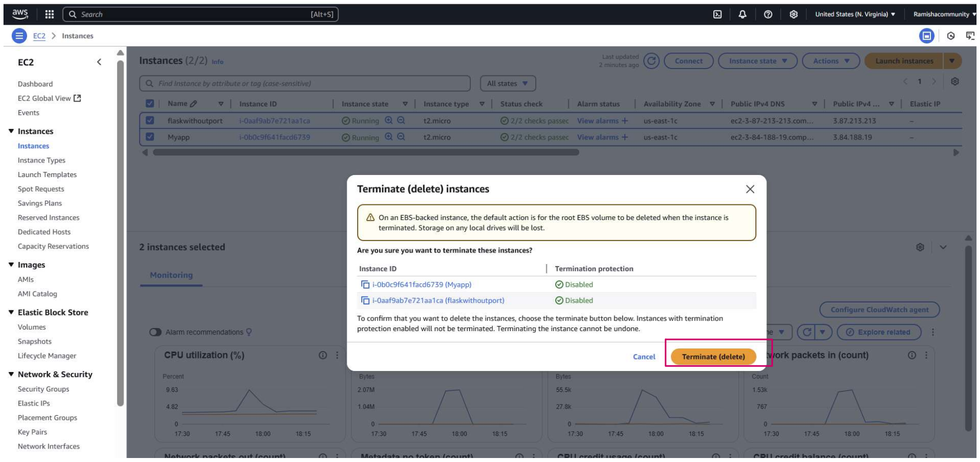
If you are practicing delete it



☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	...blic IPv4 ...	Elastic IP
☑	flaskwithoutport	i-0aaf9ab7e721aa1ca	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c	37.213.213	–
☑	Myapp	i-0b0c9f641facd6739	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c	34.188.19	–



☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	...blic IPv4 ...	Elastic IP
☑	flaskwithoutport	i-0aaf9ab7e721aa1ca	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c	37.213.213	–
☑	Myapp	i-0b0c9f641facd6739	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c	34.188.19	–



Terminate (delete) instances

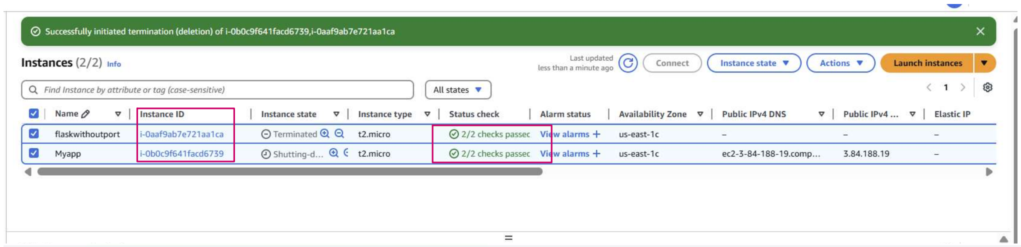
⚠ On an EBS-backed instance, the default action is for the root EBS volume to be deleted when the instance is terminated. Storage on any local drives will be lost.

Are you sure you want to terminate these instances?

Instance ID	Termination protection
i-0b0c9f641facd6739 (Myapp)	Disabled
i-0aaf9ab7e721aa1ca (flaskwithoutport)	Disabled

To confirm that you want to delete the instances, choose the terminate button below. Instances with termination protection enabled will not be terminated. Terminating the instance cannot be undone.

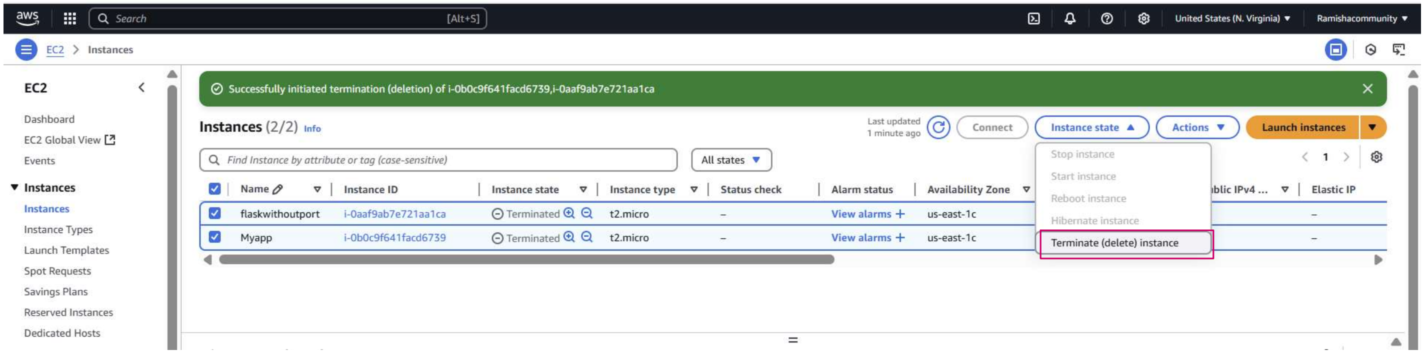
Cancel Terminate (delete)



Successfully initiated termination (deletion) of i-0b0c9f641facd6739,i-0aaf9ab7e721aa1ca

Instances (2/2) Info
Last updated less than a minute ago
Connect
Instance state ▼
Actions ▼
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states ▼

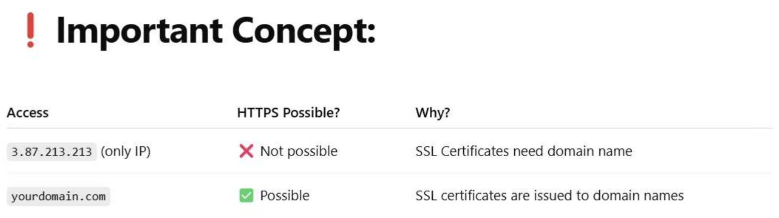
☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
☑	flaskwithoutport	i-0aaf9ab7e721aa1ca	Terminated	t2.micro	2/2 checks passec	View alarms +	us-east-1c	–	–	–
☑	Myapp	i-0b0c9f641facd6739	Shutting-d...	t2.micro	2/2 checks passec	View alarms +	us-east-1c	ec2-3-84-188-19.comp...	3.84.188.19	–



Successfully initiated termination (deletion) of i-0b0c9f641facd6739,i-0aaf9ab7e721aa1ca

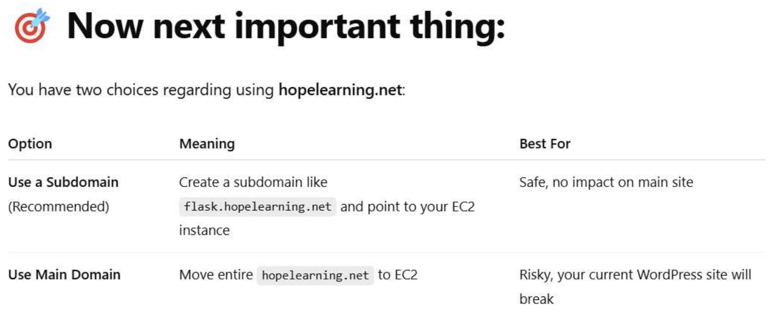
☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	...blic IPv4 ...	Elastic IP
☑	flaskwithoutport	i-0aaf9ab7e721aa1ca	Terminated	t2.micro	–	View alarms +	us-east-1c		–
☑	Myapp	i-0b0c9f641facd6739	Terminated	t2.micro	–	View alarms +	us-east-1c		–

To get into https:



❗ Important Concept:

Access	HTTPS Possible?	Why?
`3.87.213.213` (only IP)	❌ Not possible	SSL Certificates need domain name
`yourdomain.com`	✅ Possible	SSL certificates are issued to domain names



🎯 Now next important thing:

You have two choices regarding using **hopelearning.net**:

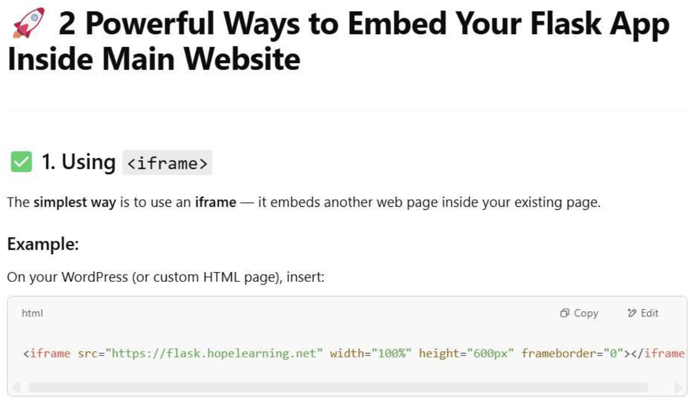
Option	Meaning	Best For
Use a Subdomain (Recommended)	Create a subdomain like `flask.hopelearning.net` and point to your EC2 instance	Safe, no impact on main site
Use Main Domain	Move entire `hopelearning.net` to EC2	Risky, your current WordPress site will break

flask.hopelearning.net
api.hopelearning.net
ec2.hopelearning.net

```
sudo yum install epel-release -y
sudo yum install certbot python3-certbot-nginx -y
sudo certbot --nginx -d flask.hopelearning.net
```

https://flask.hopelearning.net/

To run gunicorn in backend service(Production Level)



🚀 2 Powerful Ways to Embed Your Flask App Inside Main Website

✅ 1. Using `<iframe>`

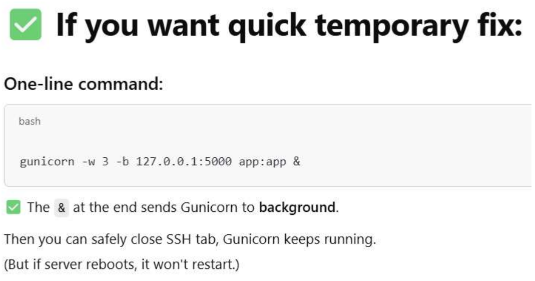
The **simplest way** is to use an **iframe** — it embeds another web page inside your existing page.

Example:

On your WordPress (or custom HTML page), insert:

```html
<iframe src="https://flask.hopelearning.net" width="100%" height="600px" frameborder="0"></iframe
```

```
gunicorn -w 3 -b 127.0.0.1:5000 app:app &
```



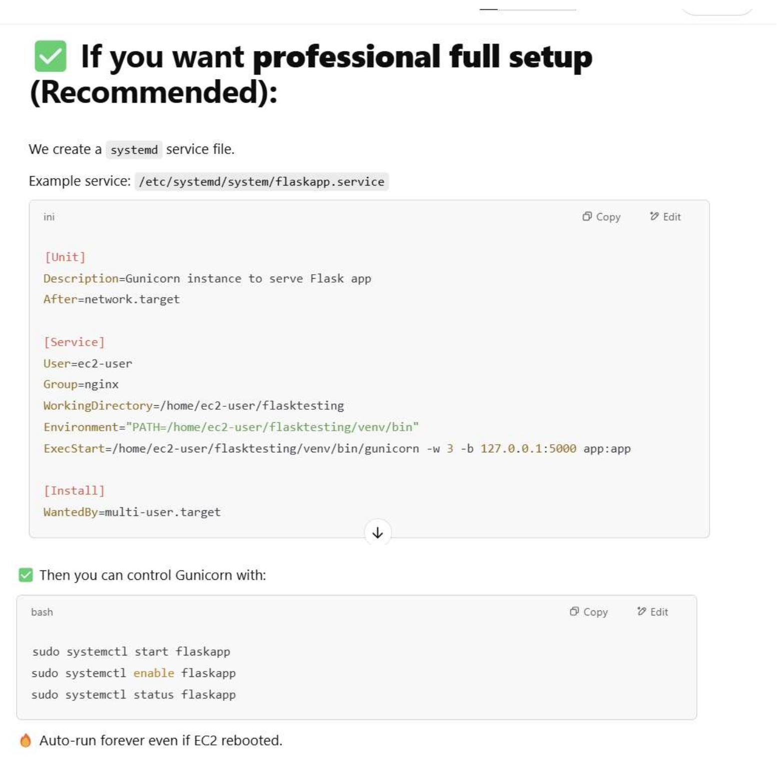
✅ If you want quick temporary fix:

One-line command:

```bash
gunicorn -w 3 -b 127.0.0.1:5000 app:app &
```

✅ The `&` at the end sends Gunicorn to **background**.

Then you can safely close SSH tab, Gunicorn keeps running.
(But if server reboots, it won't restart.)



✅ If you want **professional full setup** (Recommended):

We create a `systemd` service file.

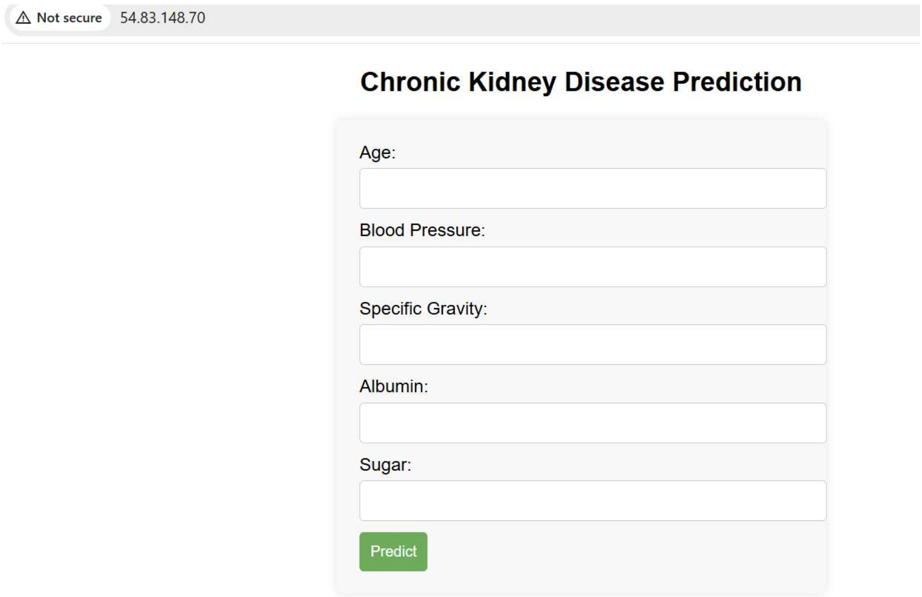
Example service: `/etc/systemd/system/flaskapp.service`

```ini
[Unit]
Description=Gunicorn instance to serve Flask app
After=network.target

[Service]
User=ec2-user
Group=nginx
WorkingDirectory=/home/ec2-user/flasktesting
Environment="PATH=/home/ec2-user/flasktesting/venv/bin"
ExecStart=/home/ec2-user/flasktesting/venv/bin/gunicorn -w 3 -b 127.0.0.1:5000 app:app

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

✅ Then you can control Gunicorn with:

```bash
sudo systemctl start flaskapp
sudo systemctl enable flaskapp
sudo systemctl status flaskapp
```

🔥 Auto-run forever even if EC2 rebooted.



⚠ Not secure 54.83.148.70

Chronic Kidney Disease Prediction

Age:

Blood Pressure:

Specific Gravity:

Albumin:

Sugar:

To kill previous Gunicorn use below comment

sudo lsof -i :5000
Sudo kill -9 pid
sudo fuser -k 5000/tcp
To troubleshoot nginx issue blow command

sudo systemctl restart nginx
Sudo nginx -t