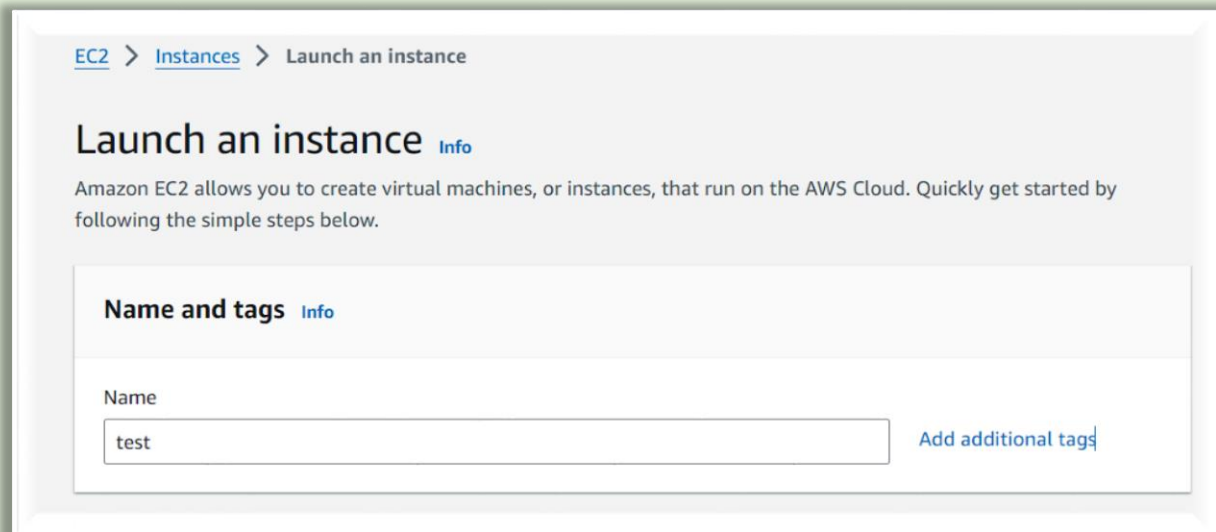


Name: - Harshal Pawar

Deploy Nginx on AWS EC2 with the custom directory.

Launch an EC2 Instance

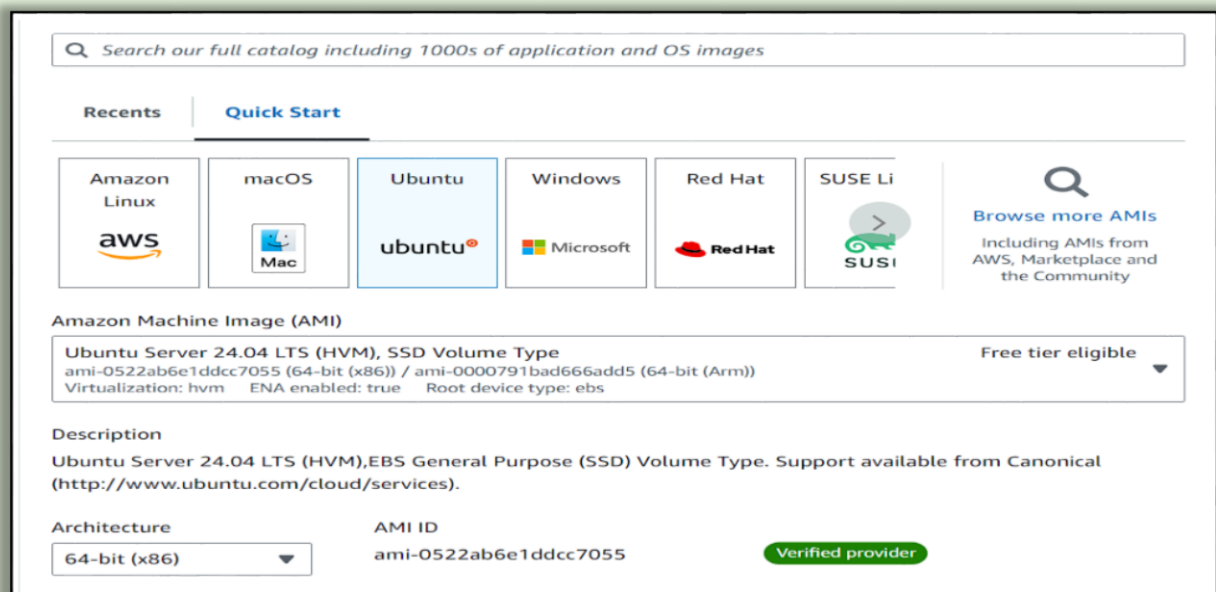
Log in to your AWS Management Console and navigate to EC2.



The screenshot shows the 'Launch an instance' page in the AWS Management Console. The breadcrumb navigation at the top reads 'EC2 > Instances > Launch an instance'. The main heading is 'Launch an instance' with an 'Info' link. Below the heading is a descriptive paragraph: '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.' A section titled 'Name and tags' with an 'Info' link contains a 'Name' label and a text input field with the value 'test'. To the right of the input field is a link 'Add additional tags'.

Launch a new instance:

Choose Ubuntu as the AMI.



The screenshot shows the 'Quick Start' section of the AWS Management Console's AMI catalog. At the top is a search bar with the placeholder text 'Search our full catalog including 1000s of application and OS images'. Below the search bar are two tabs: 'Recents' and 'Quick Start'. Under the 'Quick Start' tab, there are several AMI cards for 'Amazon Linux', 'macOS', 'Ubuntu', 'Windows', 'Red Hat', and 'SUSE Li'. The 'Ubuntu' card is highlighted. To the right of these cards is a link 'Browse more AMIs' with a magnifying glass icon, followed by the text 'Including AMIs from AWS, Marketplace and the Community'. Below the AMI cards, the 'Amazon Machine Image (AMI)' section displays details for 'Ubuntu Server 24.04 LTS (HVM), SSD Volume Type'. The details include the AMI ID 'ami-0522ab6e1ddcc7055 (64-bit (x86)) / ami-0000791bad666add5 (64-bit (Arm))', virtualization type 'hvm', ENA enabled status 'true', and root device type 'ebs'. A 'Free tier eligible' badge is visible. The 'Description' section states 'Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services)'. The 'Architecture' section shows a dropdown menu set to '64-bit (x86)'. The 'AMI ID' is 'ami-0522ab6e1ddcc7055', and a 'Verified provider' badge is present.

Select an appropriate instance type (e.g., t2.micro for free tier).

Create or choose an existing key pair for SSH access.

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t2.micro

Free tier eligible

Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0124 USD per Hour
On-Demand Windows base pricing: 0.017 USD per Hour
On-Demand RHEL base pricing: 0.0268 USD per Hour
On-Demand SUSE base pricing: 0.0124 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*

ztest

▼

↻

[Create new key pair](#)

▼ Network settings [Info](#)

Edit

Network [Info](#)

vpc-0150c7a32f58644ec

Subnet [Info](#)

No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)

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

Configure Storage and click on Launch Instance.

The screenshot shows the 'Configure storage' panel on the left and the 'Launch instance' panel on the right. The 'Configure storage' panel is titled 'Configure storage' with a 'v' icon and 'Info' link. It shows '1x 8 GiB gp3' for the root volume, which is 'Not encrypted'. A blue banner states: 'Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage'. Below this is an 'Add new volume' button. A note says: 'The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance'. Another note says: 'Click refresh to view backup information. The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.' At the bottom, it shows '0 x File systems' and an 'Edit' button. The right panel shows 'Number of instances' set to '1'. It lists the 'Software Image (AMI)' as 'Canonical, Ubuntu, 24.04, amd64...read more' with ID 'ami-0522ab6e1ddc7055'. The 'Virtual server type (instance type)' is 't2.micro'. The 'Firewall (security group)' is 'New security group'. The 'Storage (volumes)' section shows '1 volume(s) - 8 GiB'. A blue banner at the bottom of the right panel says: 'Free tier: In your first year'. At the very bottom are 'Cancel', 'Launch instance' (orange), and 'Review commands' buttons.

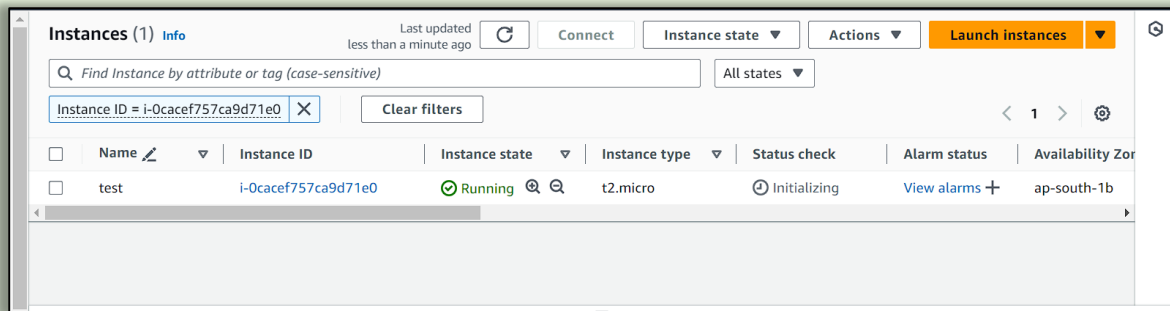
Prerequisites

An AWS EC2 instance (Ubuntu) with security group rules allowing inbound traffic on port 81 and SSH (port 22).

The screenshot shows the 'Edit inbound rules' page for a security group. The title is 'Edit inbound rules' with an 'Info' link. A subtitle says: 'Inbound rules control the incoming traffic that's allowed to reach the instance.' The main section is titled 'Inbound rules' with an 'Info' link. It contains a table with columns: 'Security group rule ID', 'Type', 'Protocol', 'Port range', 'Source', and 'Description - optional'. There are three rules listed: 1. Rule ID 'sgr-0e913fbd61b2b8195', Type 'HTTP', Protocol 'TCP', Port range '80', Source 'Cust...', Description empty. 2. Rule ID 'sgr-0c3e0bf8df0796ef0', Type 'SSH', Protocol 'TCP', Port range '22', Source 'Cust...', Description empty. 3. Rule ID '-', Type 'Custom TCP', Protocol 'TCP', Port range '81', Source 'Any...', Description empty. Each rule has a 'Delete' button. Below the table is an 'Add rule' button. At the bottom right are 'Cancel', 'Preview changes', and 'Save rules' (orange) buttons.

Now save rules.

Click on connect Instance.



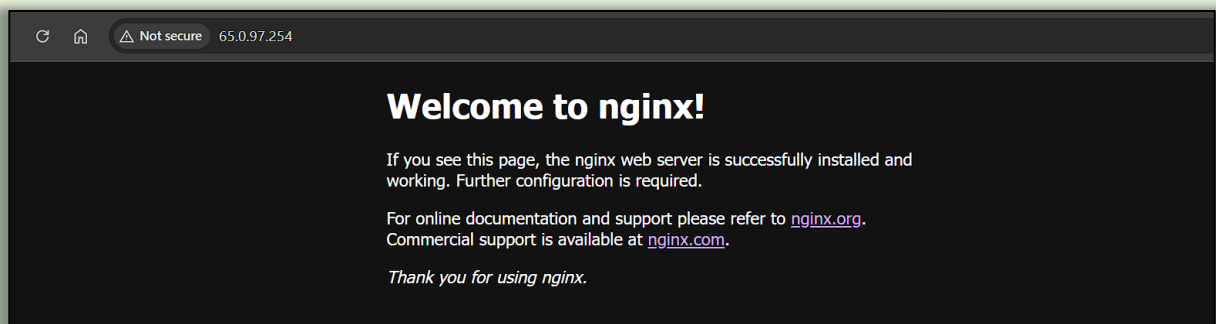
Update Package list.

```
ubuntu@ip-172-31-9-223:~$ sudo apt update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:6 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
```

Install Nginx server.

```
ubuntu@ip-172-31-9-223:~$ sudo apt install nginx
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  nginx-common
Suggested packages:
  fcgiwrap nginx-doc ssl-cert
The following NEW packages will be installed:
  nginx nginx-common
```

Try to access on default port 80 using public IP, following page will display.



“Welcome to nginx” mean we have successfully install nginx server on Ubuntu.

Create Custom directory mkdir /tutorial on location /var/www, and create index.html file inside custom directory.

```
ubuntu@ip-172-31-9-223:/var/www$ sudo mkdir tutorial
ubuntu@ip-172-31-9-223:/var/www$ cd tutorial/
ubuntu@ip-172-31-9-223:/var/www/tutorial$ sudo vim index.html
```

Paste the following to the index.html file:

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>Hello, Nginx!</title>
</head>
<body>
  <h1>Hello, Nginx!</h1>
  <p>We have just configured our Nginx web server on Ubuntu Server!</p>
</body>
</html>
~
~
~
~
```

Save this file. In next step we are going to set up virtual host to make Nginx use pages from this location.

Setting up virtual host

To set up virtual host, we need to create file in /etc/nginx/sites-enabled/ directory.

```
ubuntu@ip-172-31-9-223:~$ cd /etc/nginx/sites-enabled/
ubuntu@ip-172-31-9-223:/etc/nginx/sites-enabled$ sudo vim tutorial
```

For this tutorial, we will make our site available on 81 port, not the standard 80 port.

```
server {
    listen 81;
    listen [::]:81;

    server_name 65.0.97.254;

    root /var/www/tutorial;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

Root is a directory where we have placed our .html file. Index is used to specify file available when visiting root directory of site.

server_name is our public IP, now save and exit the file.

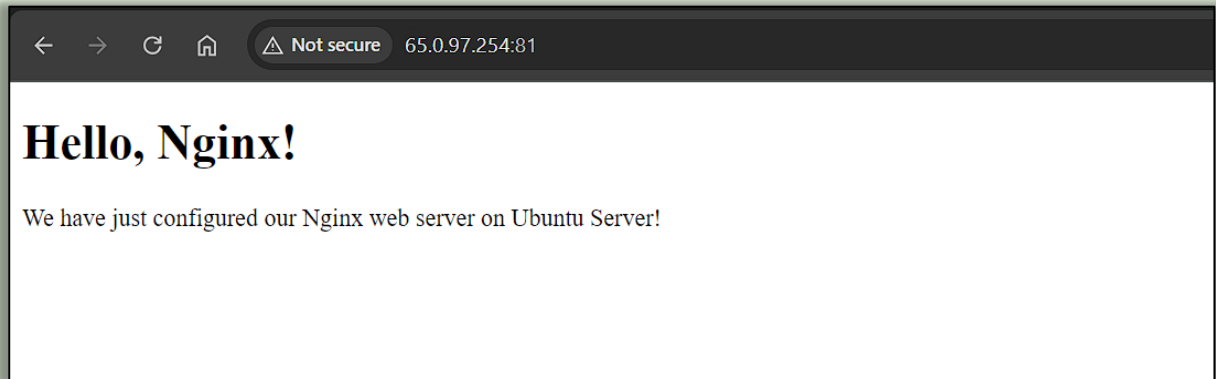
Activating virtual host and testing results

To make our site working, simply restart, enable and check status of nginx service.

```
ubuntu@ip-172-31-9-223:/etc/nginx/sites-enabled$ sudo systemctl restart nginx
ubuntu@ip-172-31-9-223:/etc/nginx/sites-enabled$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nginx
ubuntu@ip-172-31-9-223:/etc/nginx/sites-enabled$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
   Active: active (running) since Thu 2024-09-19 05:19:56 UTC; 24s ago
     Docs: man:nginx(8)
  Main PID: 2585 (nginx)
    Tasks: 2 (limit: 1130)
   Memory: 1.7M (peak: 1.9M)
      CPU: 10ms
   CGroup: /system.slice/nginx.service
           └─2585 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             └─2586 "nginx: worker process"

Sep 19 05:19:56 ip-172-31-9-223 systemd[1]: Starting nginx.service - A high performance web server and a reverse proxy server...
Sep 19 05:19:56 ip-172-31-9-223 systemd[1]: Started nginx.service - A high performance web server and a reverse proxy server.
ubuntu@ip-172-31-9-223:/etc/nginx/sites-enabled$
```

Access Website.



Congratulations! Everything works as it should. We have just configured Nginx web server.