

# ASSIGNMENT-12

**Problem Statement:** Deploy a project from GitHub to EC2 without using Port.

## Procedures:

1. Sign-in to AWS console.
2. Go to the EC2 dashboard. Now go to the instances page.
3. Click on the create new instance button.
4. Now create an EC2 server using the Security Group created earlier and enter the user data  
(Refer to Ass10)

### Launch an instance [Info](#)

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

debser1

[Add additional tags](#)

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

S

>

[Browse more AMIs](#)

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type

Free tier eligible

ami-02eb7a4783e7e9317 (64-bit (x86)) / ami-0a5dcff6fb7af3fc9 (64-bit (Arm))

Virtualization: hvm   ENA enabled: true   Root device type: ebs

Description

Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2023-03-25

Architecture

AMI ID

64-bit (x86)

ami-02eb7a4783e7e9317

Verified provider

#### ▼ Instance type [Info](#)

Instance type

t2.micro

Free tier eligible

Family: t2   1 vCPU   1 GiB Memory   Current generation: true

On-Demand Linux pricing: 0.0124 USD per Hour

On-Demand Windows pricing: 0.017 USD per Hour

On-Demand RHEL pricing: 0.0724 USD per Hour

On-Demand SUSE pricing: 0.0124 USD per Hour

☐ All generations

[Compare instance types](#)

#### ▼ 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*

debkey2

[Create new key pair](#)

#### ▼ Network settings [Info](#)

Network [Info](#)

vpc-0a33deec3fd6dc096

Subnet [Info](#)

No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)

Enable

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

Security groups [Info](#)

Select security groups

mysec1 sg-0493398d43b761e55 X

VPC: vpc-0a33deec3fd6dc096

[Compare security group rules](#)

▼ **Advanced details** [Info](#)

User data - optional [Info](#)  
Enter user data in the field.

```
#!/bin/bash
apt-get update
apt-get install -y nginx
systemctl start nginx
systemctl enable nginx
apt-get install -y git
curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -
apt-get install -y nodejs
git clone https://github.com/DebrupPramanik/myRepoV1.git
cd myRepoV1
npm install
node index.js
```

☐ User data has already been base64 encoded

[Launch instance](#)

[Review commands](#)

5. Create the instance and click on the instance after creation.

**Instances (1)** [Info](#) [Refresh](#) [Connect](#) [Instance state](#) [Actions](#) [Launch instances](#)

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input type="checkbox"/>	debser1	i-0204a46d8ffd384cb	Running	t2.micro	Initializing	No alarms	ap-south-1a	ec2-65-2-169-22

6. Copy the public IPv4 address and paste it in another browser. The nginx homepage will show up.

**Instance summary for i-0204a46d8ffd384cb (debser1)** [Info](#) [Refresh](#) [Connect](#) [Instance state](#) [Actions](#)

Updated less than a minute ago

<b>Instance ID</b> i-0204a46d8ffd384cb (debser1)	<b>Public IPv4 address</b> [Redacted] <a href="#">open address</a>	<b>Private IPv4 addresses</b> 172.31.38.17
<b>IPv6 address</b> -	<b>Instance state</b> Running	<b>Public IPv4 DNS</b> ec2-65-2-169-220.ap-south-1.compute.amazonaws.com   <a href="#">open address</a>
<b>Hostname type</b> IP name: ip-172-31-38-17.ap-south-1.compute.internal	<b>Private IP DNS name (IPv4 only)</b> ip-172-31-38-17.ap-south-1.compute.internal	



## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](https://nginx.org).  
Commercial support is available at [nginx.com](https://nginx.com).

Thank you for using nginx.

Our server is working perfectly. Note, in previous assignments we used to connect to our project webpages using port no. However, in this exercise we are going to access our project webpage without using any port no.

7. Copy the Public IPv4 address of the server instance and use this to connect it to the server using Bitwise SSH client. (Refer Ass7)

**Default profile**

Save profile as

Bitwise SSH Server Control Panel

New terminal console

Login Options Terminal RDP SFTP Services C2S S2C SSH Notes About

**Server**

Host [Redacted]

Port [Redacted] ☐ Enable obfuscation

Obfuscation keyword [Redacted]

**Authentication**

Username ubuntu

Initial method publickey

Client key Global 2

Passphrase [Redacted]

Elevation Default

**Kerberos**

SPN [Redacted]

☐ GSS/Kerberos key exchange

☐ Request delegation

☒ gssapi-keyex authentication

8. Now open the terminal in Bitvise.

```
ubuntu@65.2.169.220:22 - Bitvise xterm - ubuntu@ip-172-31-38-17: ~
Last login: Wed May 10 12:14:04 2023 from 150.129.133.232
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-38-17:~$
```

9. Enter the following commands in it.

→ pwd

```
ubuntu@ip-172-31-38-17:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-38-17:~$
```

(To check current directory)

→ cd /

```
ubuntu@ip-172-31-38-17:~$ cd /
```

(To go to root folder)

→ pwd

```
ubuntu@ip-172-31-38-17:/$ pwd
/
```

→ cd /etc/nginx/sites-available/

```
ubuntu@ip-172-31-38-17:/$ cd /etc/nginx/sites-available/
ubuntu@ip-172-31-38-17:/etc/nginx/sites-available$
```

(To open the sites-available directory under nginx)

→ sudo nano default

```
ubuntu@ip-172-31-38-17:/etc/nginx/sites-available$ sudo nano default
GNU nano 6.2 default
##
# You should look at the following URL's in order to grasp a solid understanding
# of Nginx configuration files in order to fully unleash the power of Nginx.
# https://www.nginx.com/resources/wiki/start/
# https://www.nginx.com/resources/wiki/start/topics/tutorials/config_pitfalls/
# https://wiki.debian.org/Nginx/DirectoryStructure
#
# In most cases, administrators will remove this file from sites-enabled/ and
# leave it as reference inside of sites-available where it will continue to be
# updated by the nginx packaging team.
#
# This file will automatically load configuration files provided by other
# applications, such as Drupal or Wordpress. These applications will be made
# available underneath a path with that package name, such as /drupal8.
#
# Please see /usr/share/doc/nginx-doc/examples/ for more detailed examples.
##

# Default server configuration
#
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    # SSL configuration
    #
    # listen 443 ssl default_server;
    # listen [::]:443 ssl default_server;
    #
    # Note: You should disable gzip for SSL traffic.
    # See: https://bugs.debian.org/773332

```

(To open the default file in the nano editor)

10. After opening the default file in the nano editor, search for the location / {}. It should be after server\_name\_;

```
server_name _;

location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    try_files $uri $uri/ =404;
}
```

11. Comment out the location block and each and every line inside the block.

```
server_name _;

#location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    #try_files $uri $uri/ =404;
#}
```

12. Now paste the following code just under the closing curly bracket.

```
location / {
    proxy_pass http://localhost:4000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'Upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
}
```

```
server_name _;

#location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    #try_files $uri $uri/ =404;
#}
location / {
    proxy_pass http://localhost:4000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'Upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
}
```

13. Now save it by Ctrl+X and exit nano editor.

14. You will be reverted back to the terminal. Type the following command.....

➔ **sudo systemctl restart nginx**

```
ubuntu@ip-172-31-2-192:/etc/nginx/sites-available$ sudo systemctl start nginx
```

15. Now paste the public IPv4 address in your browser. Now press Enter. Our project page will show up without entering our port no.



**We have successfully deployed a project from GitHub to EC2 without using port.**