

Name – Ruthik Jadhav

Roll No – 321087

Gr no - 22120243

Division - A3

Assignment No. 3

Title: Deploy Web application on AWS Cloud (or any cloud)(PHP/Python/Node js any application) using EC2 instance.

Theory:

What is cloud computing?

Cloud computing is a virtualization-based technology that allows us to create, configure, and customize applications via an internet connection. The cloud technology includes a development platform, hard disk, software application, and database.

The term cloud refers to a network or the internet. It is a technology that uses remote servers on the internet to store, manage, and access data online rather than local drives. The data can be anything such as files, images, documents, audio, video, and more.

Developing new applications and services

- Storage, back up, and recovery of data
- Hosting blogs and websites
- Delivery of software on demand
- Analysis of data
- Streaming videos and audios

Characteristics of Cloud Computing:

The characteristics of cloud computing are given below:

1) Agility

The cloud works in a distributed computing environment. It shares resources among users and works very fast.

2) High availability and reliability

The availability of servers is high and more reliable because the chances of infrastructure failure are minimum.

3) High Scalability

Cloud offers "on-demand" provisioning of resources on a large scale, without having engineers for peak loads.

4) Multi-Sharing

With the help of cloud computing, multiple users and applications can work more efficiently with cost reductions by sharing common infrastructure.

5) Device and Location Independence

Cloud computing enables the users to access systems using a web browser regardless of their location or what device they use e.g. PC, mobile phone, etc. As infrastructure is off-site (typically provided by a third-party) and accessed via the Internet, users can connect from anywhere.

6) Maintenance

Maintenance of cloud computing applications is easier, since they do not need to be installed on each user's computer and can be accessed from different places. So, it reduces the cost also.

Cloud Service models and Deployment models:

Types Of Cloud Computing Services:

SaaS(Software-as-a-Service):

SaaS provides clients with ability to use software applications over the internet via subscription basis. Clients can access applications from anywhere via web.

Examples: Google Applications and Salesforce.

PaaS(Platform-as-a-Service):

PaaS provides a platform where the clients can deploy their own applications and host them. The client is free from hassles of setting up infrastructure, managing storage, servers, network etc.

Examples: Amazon Web Services and Rackspace.

IaaS(Infrastructure-as-a-Service):

The IaaS provides just the hardware and network, the clients should install and develop software and applications.

Examples: IBM, Google and Amazon Web Services.

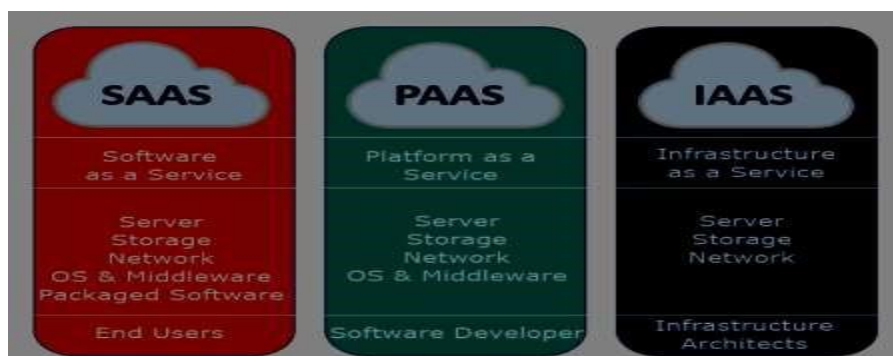


Fig. 1 Cloud Services Models

Use Cases:

SaaS(Software-as-a-Service)

Use case: Suppose you choose to take a taxi, the car agency is responsible for car finance, servicing of the car. Besides that they take care of insurance and road tax. The driver, fuel requirements is taken care as well. You just need to pay for your ride.

Similarly Software-as-a-Service provider delivers software applications over the Internet, on demand and basically on a subscription basis. You just need to pay for the service you are utilizing. Entire software and hardware stack is hosted by the provider and made available to users over the Wide Area Network(WAN) like Internet or other dedicated networks.

SaaS eliminates the need for hardware acquisition, provisioning and maintenance, as well as software licensing, installation and support. Provides scalability, flexible payments and auto updates.

Examples: Google Applications like Gmail, Google Docs.

PaaS(Platform-as-a-Service)

Use case: You plan to travel to a nearby place so decided to rent a car, then you might have to take care of fuel needs, road tolls and hire a driver as well. Rest of the work like finance of the car, car service, insurance, road tax, garage etc is responsibility of the car renting agency.

Likewise Platform-as-a-Service provider offers core computing services like storage, virtualization and network. In addition, hosts OS, middleware frameworks or other development services such as web services, database management system and SD'kits compatible with various programming languages. The service provider builds and renders a secure and optimized environment on which users can install applications and data sets.

The prime benefits of this type of service include its simplicity and convenience for users—the Platform-as-a-Service users can focus on creating and running applications rather than constructing and maintaining underlying infrastructural stack and services.

Examples: Google app Engine, Microsoft Azure, Salesforce.

IaaS(Infrastructure-as-a-Service)

Use case: You made long travel plans to a far away place so chose to **lease a car**. Here you have to worry about servicing a car, road tax, insurance and garage requirements, pay for the fuel, road tolls and hire a driver. Most of the work is done by you. The car agency takes care of just the finance related to leasing a car.

Similarly, Infrastructure-as-a-Service provider offers end users with bare computing resources like storage capacity, virtualization, networking, security and maintenance on a pay-as-you-use basis. The users are no longer concerned with location and purchase costs. Furthermore, IaaS provider supplies additional services that complement the above features like load balancing, billing details, data backup, recovery and storage.

IaaS model users handle most of the workload like installing, maintaining and managing software layers.

Example: Amazon AWS, Rack space, Flex scale and Google Cloud Platform are some well-known IaaS providers.

Cloud Deployment Models:

There are 3 fundamental Deployment Models of cloud computing: Public Cloud, Private Cloud and Hybrid Cloud.

Public cloud:

In Public Cloud model, services and infrastructure are hosted on premise of cloud provider and are provisioned for open use by general public. The end users can access the services via public network like internet. Public Cloud services are delivered mostly on demand. Popular for hosting everyday apps like email, CRM and other business support apps.

Public Cloud model offers high scalability, automated maintenance but more vulnerable to attacks due to high levels of accessibility.

Common Public Cloud providers include Amazon Web Services and Microsoft Azure. You can even check out the details of Azure with the Azure Course.

Private Cloud:

Private Cloud model provides cloud services and infrastructure exclusively to a single tenant. The tenant can control and customize it to his need. The cloud infrastructure can be monitored either by cloud provider or the tenant. Many companies are migrating their data centre's to Private Cloud to run core business fields like research, manufacturing human resource etc.

The Private Cloud model offers great levels of security and control, though cost benefits ought to be sacrificed to some extent.

Hybrid Cloud:

As the name suggests Hybrid Cloud is composition of both Public Cloud and Private Cloud infrastructure. The company can use Private Cloud to run mission critical operations and Private Cloud to run non sensitive high demand operations.

The companies using Hybrid Cloud model benefit with the security and control aspect of Private Cloud and off-hand management and cost benefits of Public Cloud.

Procedure:

First Login into AWS account and create a new EC2 instance

Step: 1] Go to EC2>Instances>Launch Instances

Launch instance

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

Launch instance ▼

Migrate a server ↗

Note: Your instances will launch in the US East (N. Virginia) Region

Service health

↻ **AWS Health Dashboard** ↗

Region
US East (N. Virginia)

Status
✔ This service is operating normally

Step: 2] Enter the instance name

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

Yash's Server

[Add additional tags](#)

Step: 3] Select Ubuntu Server

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
ami-00874d747dde814fa (64-bit (x86)) / ami-01625be155ee390e9 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible ▼

Step: 4] Select architecture for instance

Description

Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2023-01-15

Architecture
64-bit (x86) ▼

AMI ID
ami-00874d747dde814fa

Verified provider

Step: 5] select instance Type

▼ **Instance type** [Info](#)

Instance type
t2.micro Free tier eligible
Family: t2 1 vCPU 1 GiB Memory
On-Demand Windows pricing: 0.0162 USD per Hour
On-Demand SUSE pricing: 0.0116 USD per Hour
On-Demand RHEL pricing: 0.0716 USD per Hour
On-Demand Linux pricing: 0.0116 USD per Hour

[Compare instance types](#)

Step: 6] Click on create new key pair.

▼ **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](#)

Step: 7] Click on create key Pair

Generate ssh key pair for your instance. You will get an .pem file which you can use to authenticate yourself while logging to SSH.

Create key pair

Key pairs allow you to connect to your instance securely.

Enter the name of the key pair below. 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](#)

Key pair name

The name can include upto 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 (Not supported for Windows instances)

Private key file format

☒ .pem
For use with OpenSSH

☐ .ppk
For use with PuTTY

CancelCreate key pair

8] Select checkbox to allow traffic from SSH, HTTP and HTTPS in the Network Settings.

▼ Network settings Info

Edit

Network Info
vpc-06ae2038ffa2b54c5

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

We'll create a new security group called 'launch-wizard-1' 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

9] Leave other settings as default and then click on Launch instance

▼ Summary

Number of instances Info
1

Software Image (AMI)
Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2023-01-15
ami-00874d747dde814fa


Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

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

Cancel

Launch instance


Success
 Successfully initiated launch of instance (i-0456e55b4ba616f02)

▼ Launch log

Initializing requests	Succeeded
Creating security groups	Succeeded
Creating security group rules	Succeeded
Launch initiation	Succeeded

10] Check the status

Instance state = running X

Clear filters

<input type="checkbox"/>	Name ▼	Instance ID	Instance state ▼	Instance type ▼	Status check	A
<input type="checkbox"/>	Web Server	i-08c042fd7577f47be	Running	t2.micro	2/2 checks passed	N
<input type="checkbox"/>	Yash's Server	i-0c75fc52389a021	Running	t2.micro	2/2 checks passed	N

11] Copy the public key

Instance summary for i-0c75fc52389a021 (Yash's Server) info

Connect
 Instance state ▼
 Actions ▼

Instance ID
 i-0c75fc52389a021 (Yash's Server)

IPv6 address
 -

Hostname type
 IP name: ip-172-31-1-58.ec2.internal

Answer private resource DNS name
 IPv4 (A)

Auto-assigned IP address
 3.235.90.150 [Public IP]

IAM Role
 -

Public IPv4 address
 3.235.90.150 | open address

Instance state
 Running

Private IP DNS name (IPv4 only)
 ip-172-31-1-58.ec2.internal

Instance type
 t2.micro

VPC ID
 vpc-01804aedd9812db91

Subnet ID
 subnet-08a7cd602eeb61910

Private IPv4 addresses
 172.31.1.58

Public IPv4 DNS
 ec2-3-235-90-150.compute-1.amazonaws.com | open address

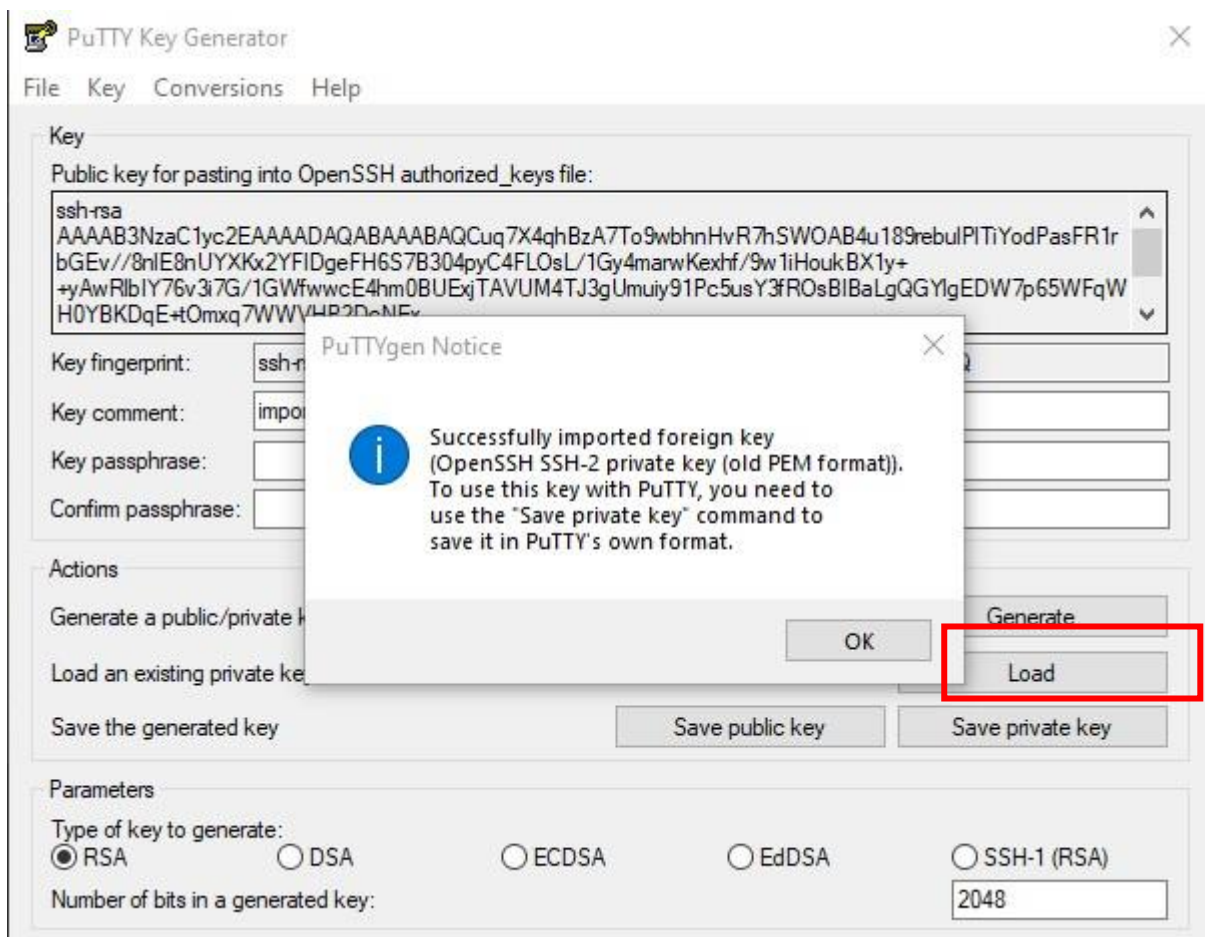
Elastic IP addresses
 -

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

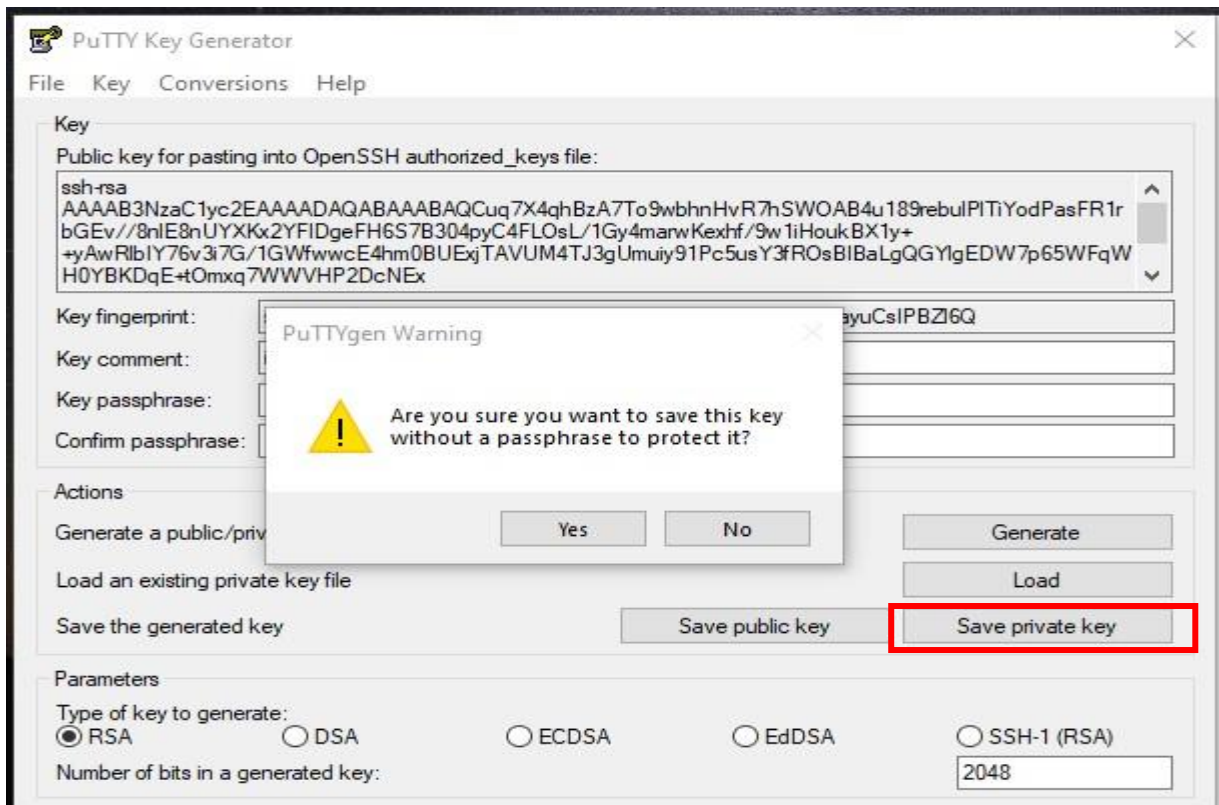
Auto Scaling Group name
 -

Steps to convert .pem file .ppk using “puttygen software”

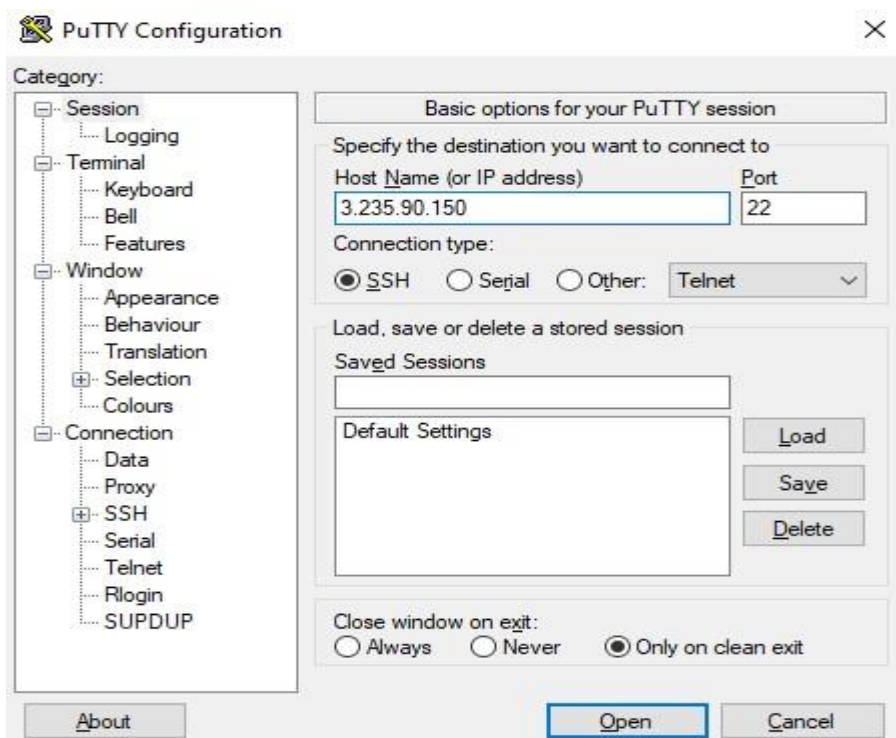
1] Open puttygen software click on “load” and select the path of your downloaded .pem file.



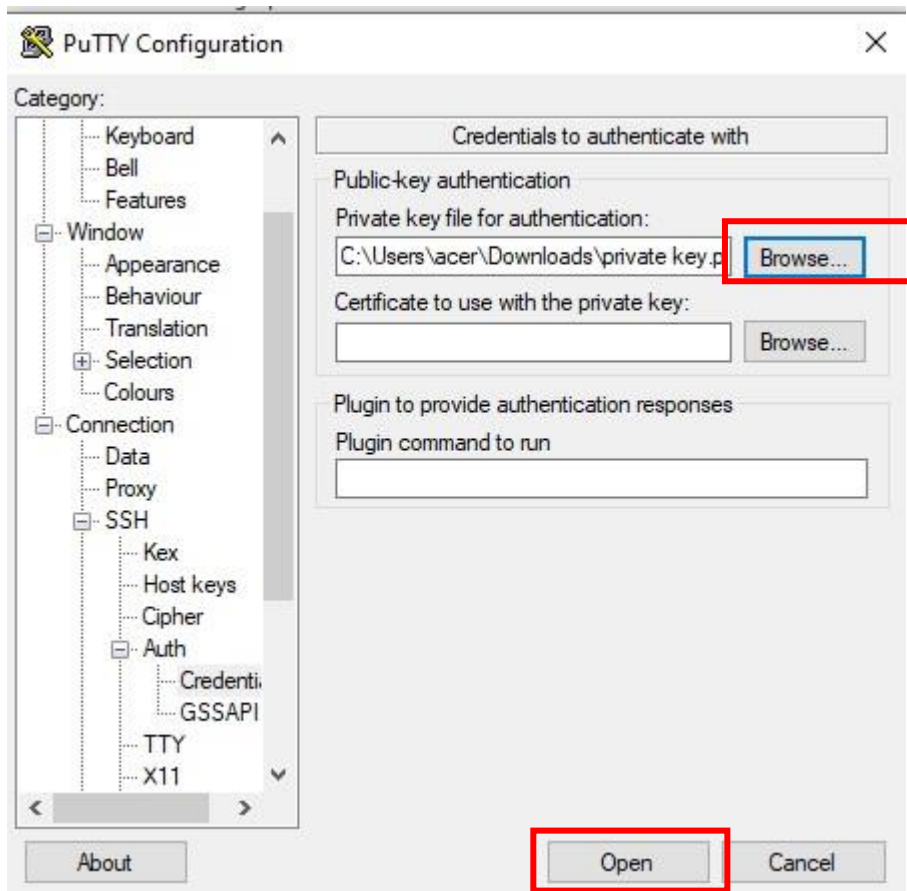
2] Click on save private key and save the key with new name



3] Now start the Putty software and copy the public ip of Ubuntu machine of aws in Host name and paste it in the PuTTY software hostname input box.



4] Click on SSH>Auth>credentials-> Browse Give the path of downloaded path of private key. And click on open. It will open a command prompt.



5] Login as Ubuntu. Now you are in Ubuntu machine running in AWS.

```
ubuntu@ip-172-31-5-48: ~  
login as: ubuntu  
Authenticating with public key "imported-openssh-key"  
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1028-aws x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
System information as of Fri Feb 10 08:29:15 UTC 2023  
  
System load:  0.0           Processes:            95  
Usage of /:   36.3% of 7.57GB Users logged in:          0  
Memory usage: 20%          IPv4 address for eth0: 172.31.5.48  
Swap usage:   0%  
  
0 updates can be applied immediately.  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
Last login: Thu Feb  9 19:41:02 2023 from 49.36.48.136  
ubuntu@ip-172-31-5-48:~$
```

Steps to Deploy Web application:

1] Make myweb directory for storing web application.

```
ubuntu@ip-172-31-5-48:~/workspace$ mkdir myweb
```

2] Clone github repository using git

```
ubuntu@ip-172-31-1-58:~/workspace/app$ git clone https://github.com/yashrajtalokar/ToDo.git  
Cloning into 'ToDo'...  
remote: Enumerating objects: 24, done.  
remote: Counting objects: 100% (24/24), done.  
remote: Compressing objects: 100% (18/18), done.  
remote: Total 24 (delta 3), reused 0 (delta 0), pack-reused 0  
Receiving objects: 100% (24/24), 5.66 KiB | 1.88 MiB/s, done.  
Resolving deltas: 100% (3/3), done.  
ubuntu@ip-172-31-1-58:~/workspace/app$
```


3] Install the required dependencies depending on your project and run web application

```
Last login: Fri Feb 10 15:18:35 2023 from 49.36.49.24
ubuntu@ip-172-31-1-58:~$ cd workspace
ubuntu@ip-172-31-1-58:~/workspace$ cd ToDo
ubuntu@ip-172-31-1-58:~/workspace/ToDo$ ls
README.md  date.js      node_modules  package.json  views
app.js     index.html  package-lock.json  public
ubuntu@ip-172-31-1-58:~/workspace/ToDo$ node app.js
```

```
ubuntu@ip-172-31-1-58:~/workspace/ToDo$ node app.js
Server is running on port 3000
```

6] Run 3.235.90.150:3000 this public IPv4 address with port number in browser and see the deployed website.

