

# Weather App

## Project Documentation

# **Contents**

- 1. Introduction**
- 2. Technology Used**
- 3. Installation Instructions**
- 4. Home Page**
- 5. Dockerizing Angular App**
- 6. Deploying on AWS EC2 Instance**

## Introduction

Weather App is Angular Based web application which can be used to get latest weather information like temperature, min and max temperature, humidity and wind. Users can search for their city weather by entering on search city text field.

## Technologies Used

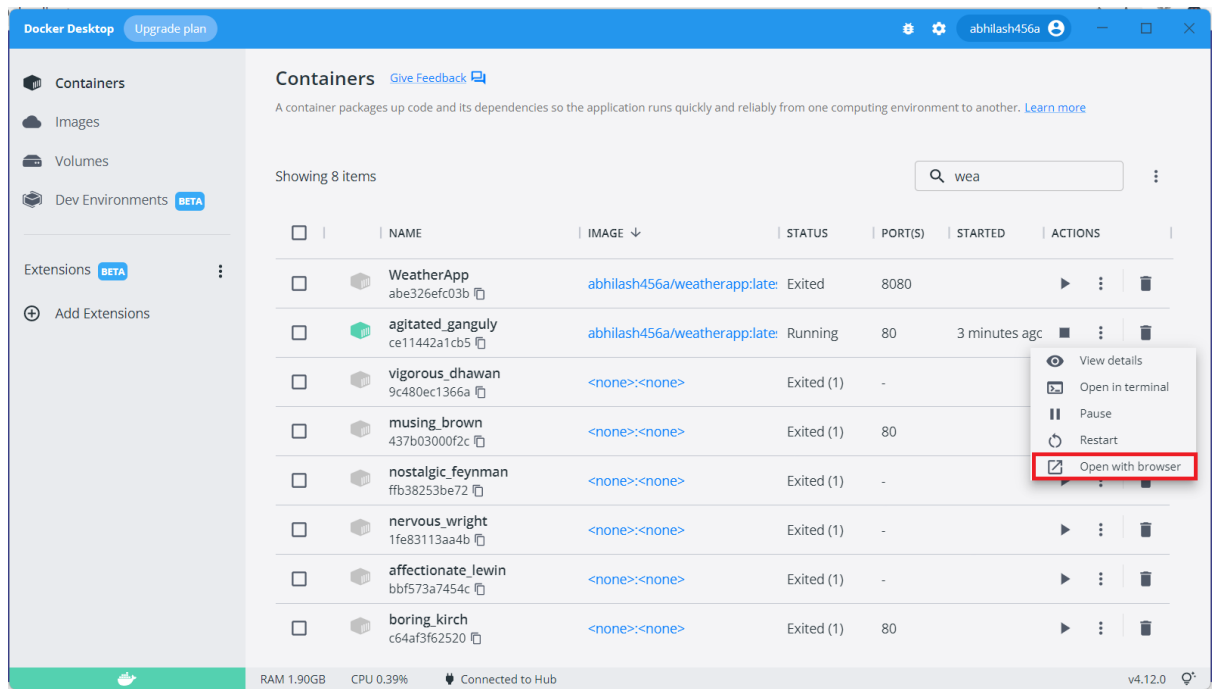
- Visual Studio
- Angular
- AWS EC2 Instance
- Docker
- Node JS
- GitHub

## Installation Instructions

Users need to install docker on their machine and run the following command

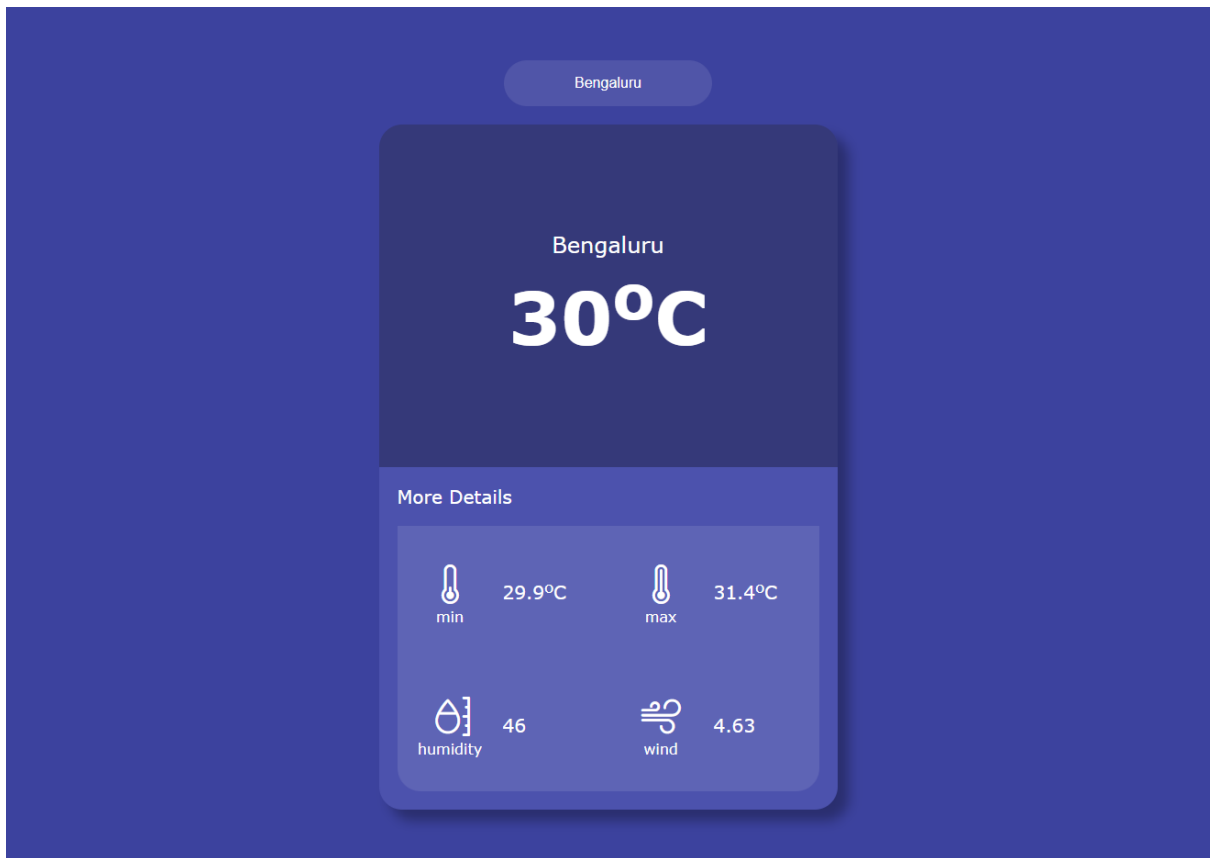
```
docker run -d -p 80:80 abhilash456a/weatherapp:latest
```

Users can open the weatherapp by going to docker desktop and selecting the container that's running and click on vertical dot menu and select open in browser option to view inside the browser.



To Download the source users can download from the github repository at <https://github.com/abhilashlegend/weatherapp.git> . After downloading into your computer. Run npm install to download the packages and dependencies. And Run ng serve to launch on the browser. The angular app is usually hosted at <http://localhost:4200/>

## Home Page



Weather App is a single page application that displays weather details. On the home page you can see the details of default city that is Bengaluru. The main temperature is 30 degrees Celsius. At the bottom there is min and max temperature of the city, humidity and wind. User can change the city by clicking on the top text field and entering.

# Dockerizing Angular App

Steps to dockerizing the angular App

1. Install Docker Desktop
2. Create Dockerfile in the project root folder.

```
1  # Use an official Node.js runtime as the base image
2  FROM node:latest as build
3
4  # Set the working directory in the container
5  WORKDIR /usr/local/app
6
7  # Copy package.json and package-lock.json to the container
8  COPY ./ /usr/local/app/
9
10 # Install application dependencies
11 RUN npm install
12
13 # Generate the build of the application
14 RUN npm run build
15
16 # Stage 2: Serve app with nginx server
17
18 # Use official nginx image as the base image
19 FROM nginx:latest
20
21 # Copy the build output to replace the default nginx contents.
22 COPY --from=build /usr/local/app/dist/weatherapp
   /usr/share/nginx/html
23
24 # Expose port 80
25 EXPOSE 80
26
```

3. Run ng build command to build the angular application
4. Use the following command to generate the Docker image for the Angular application using Dockerfile:  
Example: `docker build -t dockerhub_name/image_name:tag dockerfile_location`

*`docker build -t abhilash456a/weatherapp:latest .`*

5. Get the list of Docker images using the following command:

*Docker image ls*

6. You need to push the Docker image to Docker Hub or any container registry(AWS ECR, Azure CR) if you want to deploy the application on the Cloud server.

To push the Docker image to the Docker hub you need a Docker hub account. Once you created a Docker hub account, then log in to the Docker hub on your terminal.

*docker login*

7. Use the following command to push the Docker image to Docker Hub:

*docker push abhilash456a/weatherapp:latest*

8. Run Docker Container

Run the Angular application using the following command

*docker run -d -p 80:80 abhilash456a/weatherapp:latest*

It runs on port number 80. Access the Angular application using the IP address and port number.

<http://localhost:80/>

9. list the containers by below command->

*docker ps*

# Deploying on AWS EC2 Instance

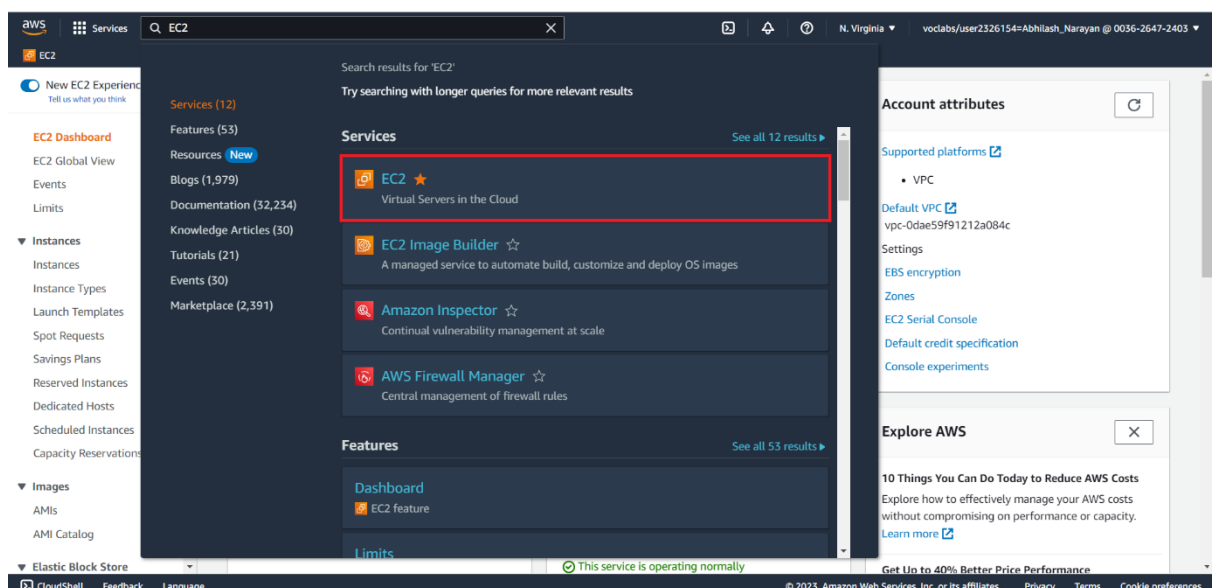
## Steps to deploy on AWS EC2 Instance

1. open <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>
2. Click on Linux/macOS or Windows as per your Operating system  
For Windows:  
Download and run the AWS CLI MSI installer for Windows (64-bit)
3. To confirm the installation, open the Start menu, search for cmd to open a command prompt window, and at the command prompt use the aws --version command.  

```
C:\> aws --version
```

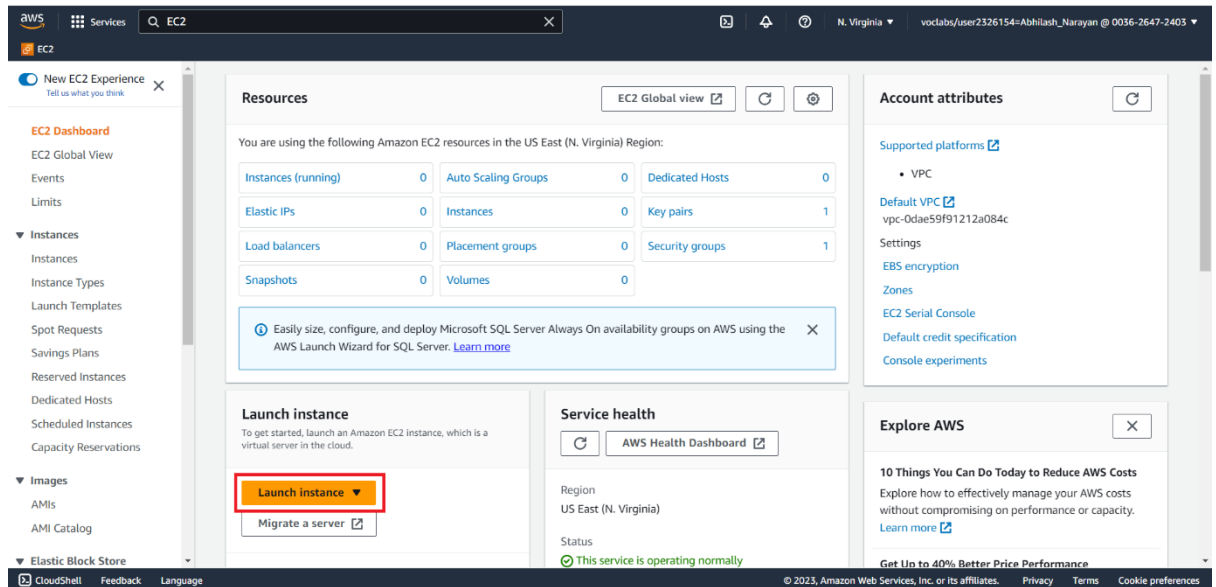
```
o/p-> aws-cli/2.10.0 Python/3.11.2 Windows/10 exe/AMD64
```

prompt/off
4. Go to AWS and login to your account and select EC2 Instance  
Click services -> EC2 (or search for it if unable to find)



5. Click EC2 dashboard->Launch instance





a. Give name as weatherapp

**Name and tags** [Info](#)

Name

weatherapp

[Add additional tags](#)

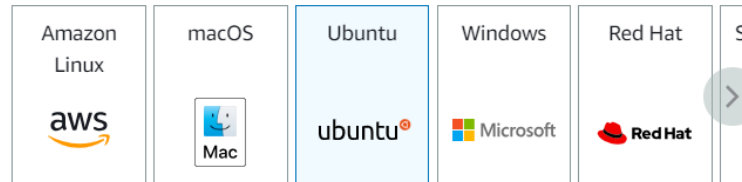
b. In Application and OS Images select Ubuntu  
Amazon Machine Image -> Ubuntu Server 22.04 LTS (HVM), SSD  
Volume Type ( Free Tier eligible)

### ▼ 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

#### Quick Start



[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-053b0d53c279acc90 (64-bit (x86)) / ami-0a0c8eebcdd6dcbd0 (64-bit (Arm))  
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible ▼

#### Description

Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2023-05-16

#### Architecture

#### AMI ID

64-bit (x86) ▼

ami-053b0d53c279acc90

Verified provider

### c. Instance type->t2.micro (free tier eligible)

### ▼ Instance type [Info](#)

#### Instance type

t2.micro

Free tier eligible

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

☐ All generations

[Compare instance types](#)

### d. Keypair (login)

### ▼ 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.

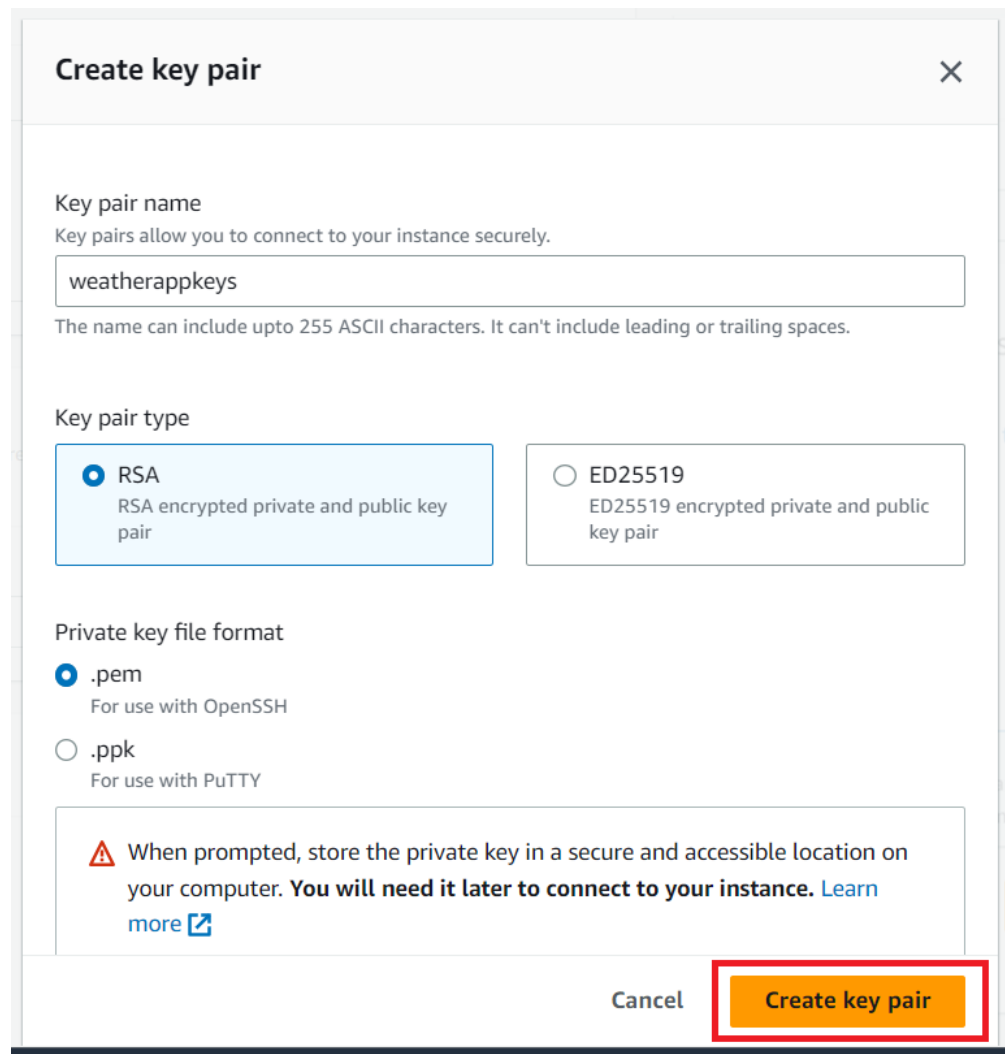
Key pair name - *required*

Select ▼

[Create new key pair](#)

A create key pair window will open.

- Enter key pair name -> weatherappkeys.
- Select Key pair type -> RSA
- Private key file format -> .pem
- Click create new Key pair



The screenshot shows the 'Create key pair' dialog box. At the top, the title is 'Create key pair' with a close button (X) on the right. Below the title, there is a section for 'Key pair name' with a text input field containing 'weatherappkeys'. A note below the field states: 'Key pairs allow you to connect to your instance securely. The name can include upto 255 ASCII characters. It can't include leading or trailing spaces.' Below this, there is a section for 'Key pair type' with two radio button options: 'RSA' (selected) and 'ED25519'. The 'RSA' option is described as 'RSA encrypted private and public key pair'. The 'ED25519' option is described as 'ED25519 encrypted private and public key pair'. Below the key pair type section, there is a section for 'Private key file format' with two radio button options: '.pem' (selected) and '.ppk'. The '.pem' option is described as 'For use with OpenSSH'. The '.ppk' option is described as 'For use with PuTTY'. At the bottom of the dialog, there is a warning icon and text: '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'. At the bottom right, there are two buttons: 'Cancel' and 'Create key pair' (highlighted with a red border).

The keys file(weatherappkeys.pem) will get downloaded in your system, you will need them later to connect to EC2 instance.

6. create a folder in c: by name weatherappkeys. Save the weatherappkeys.pem file in c:/weatherappkeys
7. Click Launch Instance(No changes required for other settings )

▼ Network settings

Info

Edit

Network

Info

vpc-0dae59f91212a084c

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

▼ Summary

Number of instances

Info

1

Software Image (AMI)

Canonical, Ubuntu, 22.04 LTS, ...read more

ami-053b0d53c279acc90

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Cancel

Launch instance

Review commands

EC2 > Instances > Launch an instance

✓ Success

Successfully initiated launch of instance (i-0153310eb88c8bf09)

▼ Launch log

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

## 8. Click on instances

EC2

New EC2 Experience

Tell us what you think

EC2 Dashboard

EC2 Global View

Events

Instances (1)

Info

Find instance by attribute or tag (case-sensitive)

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"/>	weatherapp	i-0153310eb88c8bf09	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-44-193-73-1

## 9. To connect this instance select checkbox before this row and click on connect

Instances (1/1)

Info

Find instance by attribute or tag (case-sensitive)

Connect

Instance state

Actions

Launch instances

<input checked="" type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input checked="" type="checkbox"/>	weatherapp	i-0153310eb88c8bf09	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-44-193-73-1

## Connect to instance [Info](#)

Connect to your instance i-0153310eb88c8bf09 (weatherapp) using any of these options

EC2 Instance Connect



Session Manager

**SSH client**


EC2 serial console


Instance ID

 i-0153310eb88c8bf09 (weatherapp)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is weatherappkeys.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.  
 `chmod 400 weatherappkeys.pem`
4. Connect to your instance using its Public DNS:  
 `ec2-44-193-73-144.compute-1.amazonaws.com`

Example:

 `ssh -i "weatherappkeys.pem" ubuntu@ec2-44-193-73-144.compute-1.amazonaws.com`

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

Click on SSH client copy the command  
copy the command below Example:

`ssh -i "weatherappkeys.pem" ubuntu@ec2-44-193-73-144.compute-1.amazonaws.com`

10. Open git bash(you can search for it in search area next to start menu)  
browse to location where u saved the keys i.e c:/weatherappkeys by

> cd c:

> cd weatherappkeys

To check ur current location

>pwd

```
MINGW64:/c/weatherappkeys

Abhil@abhilash MINGW64 ~
$ C:
bash: C:: command not found

Abhil@abhilash MINGW64 ~
$ cd C:

Abhil@abhilash MINGW64 /c
$ cd weatherappkeys

Abhil@abhilash MINGW64 /c/weatherappkeys
$ pwd
/c/weatherappkeys

Abhil@abhilash MINGW64 /c/weatherappkeys
$
```

11. Now paste (use right click paste as ctrl v will not work) the command copied earlier

```
ssh -i "weatherappkeys.pem" ubuntu@ec2-44-193-73-144.compute-1.amazonaws.com
```

Press enter

```
MINGW64:/c/weatherappkeys

Abhil@abhilash MINGW64 ~
$ C:
bash: C:: command not found

Abhil@abhilash MINGW64 ~
$ cd C:

Abhil@abhilash MINGW64 /c
$ cd weatherappkeys

Abhil@abhilash MINGW64 /c/weatherappkeys
$ pwd
/c/weatherappkeys

Abhil@abhilash MINGW64 /c/weatherappkeys
$ ssh -i "weatherappkeys.pem" ubuntu@ec2-44-193-73-144.compute-1.amazonaws.com
The authenticity of host 'ec2-44-193-73-144.compute-1.amazonaws.com (44.193.73.144)' can't be established.
ED25519 key fingerprint is SHA256:2QYR2BvZIANVAN8wgCL9v2d5fip7OB03XWJ4QDDzZis.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

```
MINGW64:/c/weatherappkeys

Abhil@abhilash MINGW64 ~
$ cd C:

Abhil@abhilash MINGW64 /c
$ cd weatherappkeys

Abhil@abhilash MINGW64 /c/weatherappkeys
$ pwd
/c/weatherappkeys

Abhil@abhilash MINGW64 /c/weatherappkeys
$ ssh -i "weatherappkeys.pem" ubuntu@ec2-44-193-73-144.compute-1.amazonaws.com
The authenticity of host 'ec2-44-193-73-144.compute-1.amazonaws.com (44.193.73.144)' can't be established.
ED25519 key fingerprint is SHA256:2QYR2BvZIANVAN8wgCL9v2d5fip7OB03XWJ4QDDzZis.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-193-73-144.compute-1.amazonaws.com' (ED25519)
to the list of known hosts.
Connection closed by 44.193.73.144 port 22

Abhil@abhilash MINGW64 /c/weatherappkeys
$
```

12. Again give same command i.e.

```
Abhil@abhilash MINGW64 /c/weatherappkeys
$ ssh -i "weatherappkeys.pem" ubuntu@ec2-44-193-73-144.compute-1.amazonaws.com
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-1025-aws x86_64)

12. Again give same command i.e.

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Thu Jun 15 15:16:23 UTC 2023

System load:  0.0          Processes:    114
Usage of /:   20.6% of 7.57GB   Users logged in: 0
Memory usage: 27%          IPv4 address for eth0: 172.31.8.25
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-8-25:~$
```

13. Run the following command to install NGINX

```
sudo -s - for super user
sudo apt update - to update the existing packages
sudo apt install nginx - to install the nginx web server
```

14. Check if git is installed

*git --version*

To install git in EC2 virtual machine->

*sudo apt install git -y*

To check if git is successfully installed run the command again

*git --version*

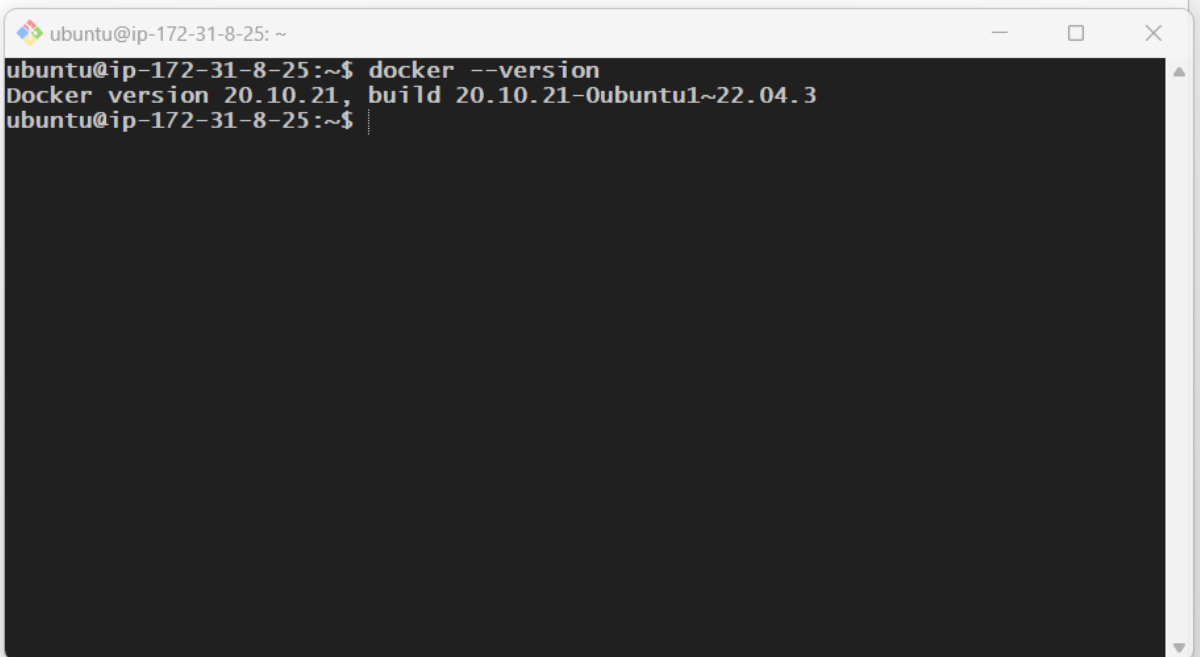
15. Install docker in this virtual EC2 machine

// install most recent package

*sudo apt install docker.io*

Check if docker was installed by running the command

*docker --version*

A terminal window with a title bar showing 'ubuntu@ip-172-31-8-25: ~'. The terminal content shows the command 'docker --version' being executed, resulting in the output 'Docker version 20.10.21, build 20.10.21-0ubuntu1~22.04.3'. The prompt 'ubuntu@ip-172-31-8-25:~\$' is visible on both lines.

```
ubuntu@ip-172-31-8-25: ~  
ubuntu@ip-172-31-8-25:~$ docker --version  
Docker version 20.10.21, build 20.10.21-0ubuntu1~22.04.3  
ubuntu@ip-172-31-8-25:~$
```

16. Start the service docker

*sudo service docker start*

17. Pull the docker image

*sudo docker pull abhilash456a/weatherapp*



```
ubuntu@ip-172-31-8-25: ~  
ubuntu@ip-172-31-8-25:~$ docker --version  
Docker version 20.10.21, build 20.10.21-0ubuntu1~22.04.3  
ubuntu@ip-172-31-8-25:~$ sudo service docker start  
ubuntu@ip-172-31-8-25:~$ sudo docker pull abhilash456a/weatherapp  
Using default tag: latest  
latest: Pulling from abhilash456a/weatherapp  
f03b40093957: Pull complete  
eed12bbd6494: Pull complete  
fa7eb8c8eee8: Pull complete  
7ff3b2b12318: Pull complete  
0f67c7de5f2c: Pull complete  
831f51541d38: Pull complete  
9dd8b0c4050e: Pull complete  
Digest: sha256:02176b9900427f81cf97221631d24a2a7dd50acfa01024237316c9906f26a640  
Status: Downloaded newer image for abhilash456a/weatherapp:latest  
docker.io/abhilash456a/weatherapp:latest  
ubuntu@ip-172-31-8-25:~$
```

18. List the images

*sudo docker images*

```
ubuntu@ip-172-31-8-25: ~  
ubuntu@ip-172-31-8-25:~$ docker --version  
Docker version 20.10.21, build 20.10.21-0ubuntu1~22.04.3  
ubuntu@ip-172-31-8-25:~$ sudo service docker start  
ubuntu@ip-172-31-8-25:~$ sudo docker pull abhilash456a/weatherapp  
Using default tag: latest  
latest: Pulling from abhilash456a/weatherapp  
f03b40093957: Pull complete  
eed12bbd6494: Pull complete  
fa7eb8c8eee8: Pull complete  
7ff3b2b12318: Pull complete  
0f67c7de5f2c: Pull complete  
831f51541d38: Pull complete  
9dd8b0c4050e: Pull complete  
Digest: sha256:02176b9900427f81cf97221631d24a2a7dd50acfa01024237316c9906f26a640  
Status: Downloaded newer image for abhilash456a/weatherapp:latest  
docker.io/abhilash456a/weatherapp:latest  
ubuntu@ip-172-31-8-25:~$ sudo docker images  
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE  
abhilash456a/weatherapp latest      1ea90565cc11  2 days ago   143MB  
ubuntu@ip-172-31-8-25:~$
```

- click instance->security tab-> click security groups link sg-04b7f86c89500d771 (launch-wizard-1)

Instances (1/1) Info

Find instance by attribute or tag (case-sensitive)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
weatherapp	i-0153310eb88c8bf09	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-44-193-73-1

Instance: i-0153310eb88c8bf09 (weatherapp)

Details Security Networking Storage Status checks Monitoring Tags

▼ Security details

IAM Role: -

Owner ID: 003626472403

Launch time: Thu Jun 15 2023 20:13:16 GMT+0530 (India Standard Time)

Security groups: sg-04b7f86c89500d771 (launch-wizard-1)

▼ Inbound rules

Go to inbound rules tab and click on edit inbound rules

Details

Security group name: launch-wizard-1

Security group ID: sg-04b7f86c89500d771

Description: launch-wizard-1 created 2023-06-15T13:57:33.500Z

VPC ID: vpc-0dae59f91212a084c

Owner: 003626472403

Inbound rules count: 1 Permission entry

Outbound rules count: 1 Permission entry

Inbound rules Outbound rules Tags

You can now check network connectivity with Reachability Analyzer

Run Reachability Analyzer

Inbound rules (1/1)

Filter security group rules

Name	Security group rule...	IP version	Type	Protocol	Port range
-	sgr-090a02be82922e6...	IPv4	SSH	TCP	22

Edit inbound rules

Click Add Rule

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-00d276679c59af3ca	Custom TCP	TCP	8081	Custom	

0.0.0.0/0

Select Custom TCP, Port Range 8081, change custom to Anywhere-IPv4  
And save changes

19. In gitbash run the container using the command  
`sudo docker run -d -p 8081:80 abhilash456a/weatherapp`

20. Check your public ip address and go to port 8081 Eg:  
<http://54.196.220.154:8081>

