Deploying a Node.js Application On AWS EC2 Server

Chapter 1

Introduction

Nodejs

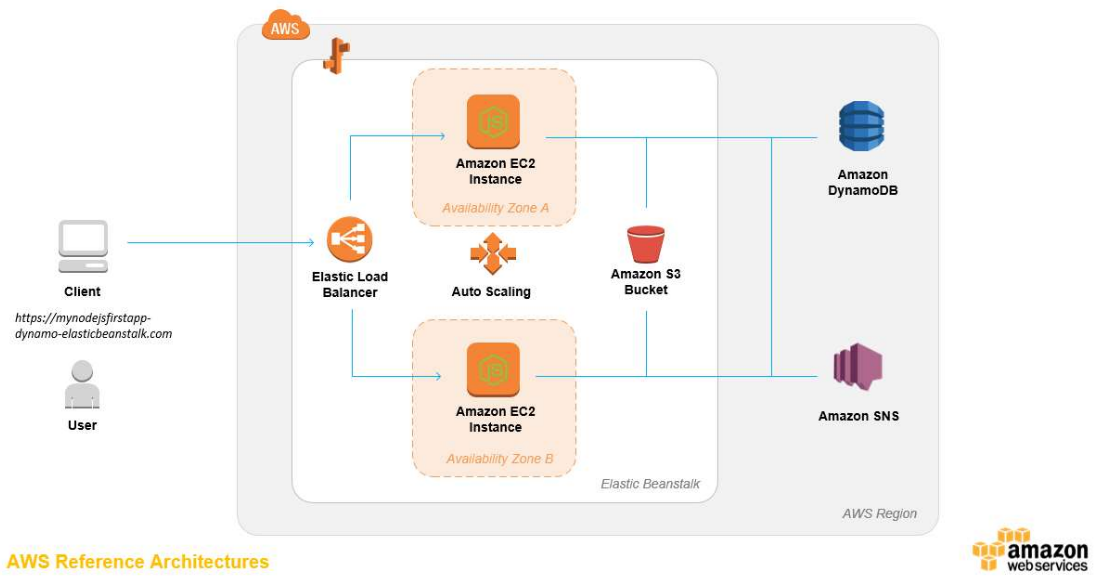
Node Js is a runtime for javascript, that is built on top of google chrome V8 engine which is writing in C++.

AWS

Amazon Web Services is very popular and reliable platform to host our application on top of their servers. There are many types of amazon's services that we can use to host web app, one of which is EC2 (Elastic Compute Service). It is free and provides a very customized configuration during the creation process. An instance can be created: a set amount of resources initialized (OS, disk space, Ram, etc).

EC2

EC2 is free and provides very customizable configurations to the end user. Favourite Linux based operating system can be selected and can all the required utilities and software on that OS can be installed.



Chapter 2

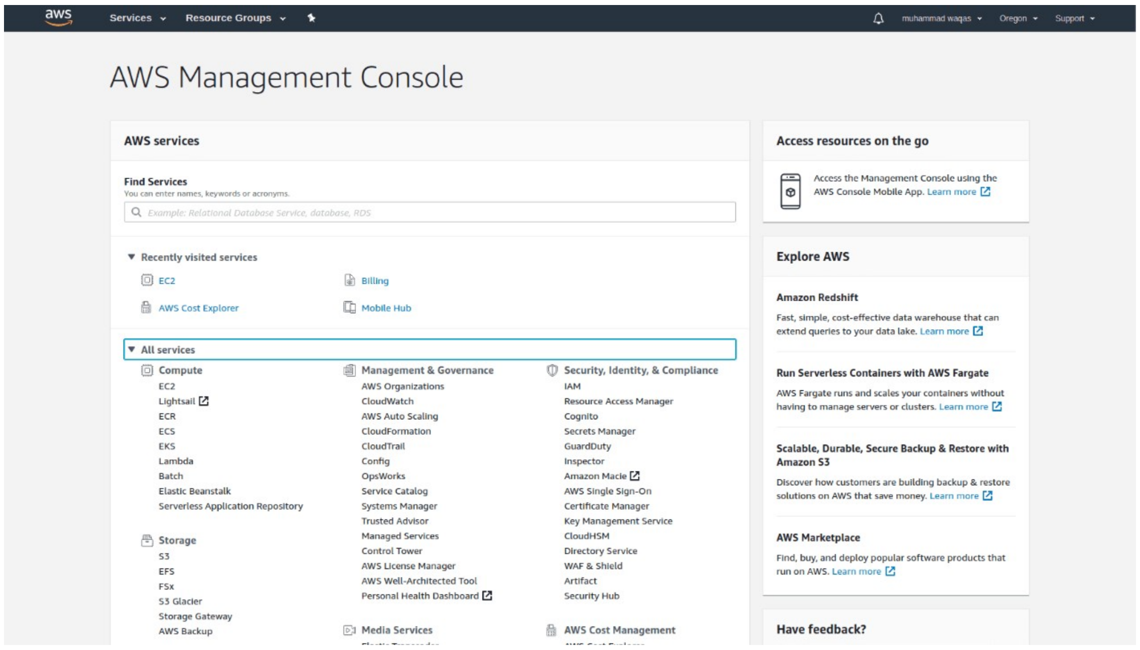
Steps to deploy nodejs application on AWS EC2 Server

1. Create account on Aws

Go to https://www.aws.com and create new account on aws. Then go to aws console. An AWS new account lets to try the free tier of Amazon EC2, Amazon S3 and Amazon DynamoDb services for 12 months.

2. Launch an EC2 instance

Now go to aws console and provide login credentials. Once logged in all the aws services under the "AWS Services" section can be seen.



Access the Management Console using the AWS Console Mobile App. Learn more

Fast, simple, cost-effective data warehouse that can extend queries to your data lake. Learn more

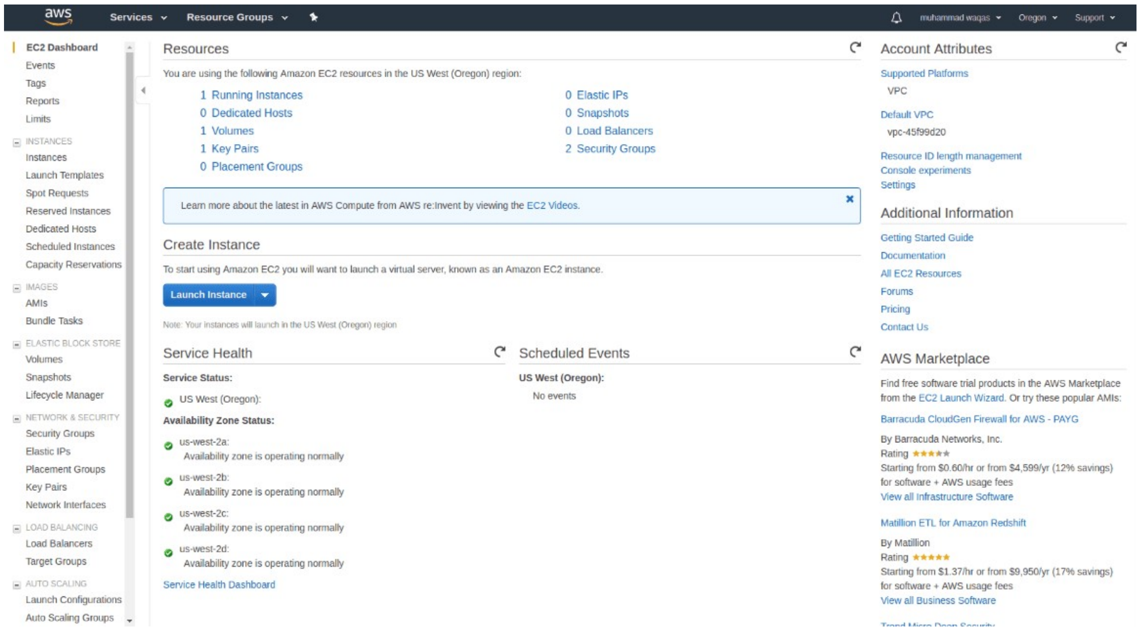
AWS Fargate runs and scales your containers without having to manage servers or clusters. Learn more

Discover how customers are building backup & restore solutions on AWS that save money. Learn more

Find, buy, and deploy popular software products that run on AWS. Learn more

- **AWS Management Console**

  Click on the EC2 Service under Computer Service, it will show EC2 Dashboard page, on this page just click on Launch Instance button under Create Instance Section.



Resources

You are using the following Amazon EC2 resources in the US West (Oregon) region:

1 Running Instances	0 Elastic IPs
0 Dedicated Hosts	0 Snapshots
1 Volumes	0 Load Balancers
1 Key Pairs	2 Security Groups
0 Placement Groups	

Learn more about the latest in AWS Compute from AWS re:Invent by viewing the EC2 Videos.

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Launch Instance

Note: Your instances will launch in the US West (Oregon) region

Service Health

Service Status:

US West (Oregon):

Availability Zone Status:

us-west-2a:
Availability zone is operating normally

us-west-2b:
Availability zone is operating normally

us-west-2c:
Availability zone is operating normally

us-west-2d:
Availability zone is operating normally

Service Health Dashboard

Scheduled Events

US West (Oregon):
No events

Account Attributes

Supported Platforms
VPC

Default VPC
vpc-45f99d20

Resource ID length management
Console experiments
Settings

Additional Information

Getting Started Guide
Documentation
All EC2 Resources
Forums
Pricing
Contact Us

AWS Marketplace

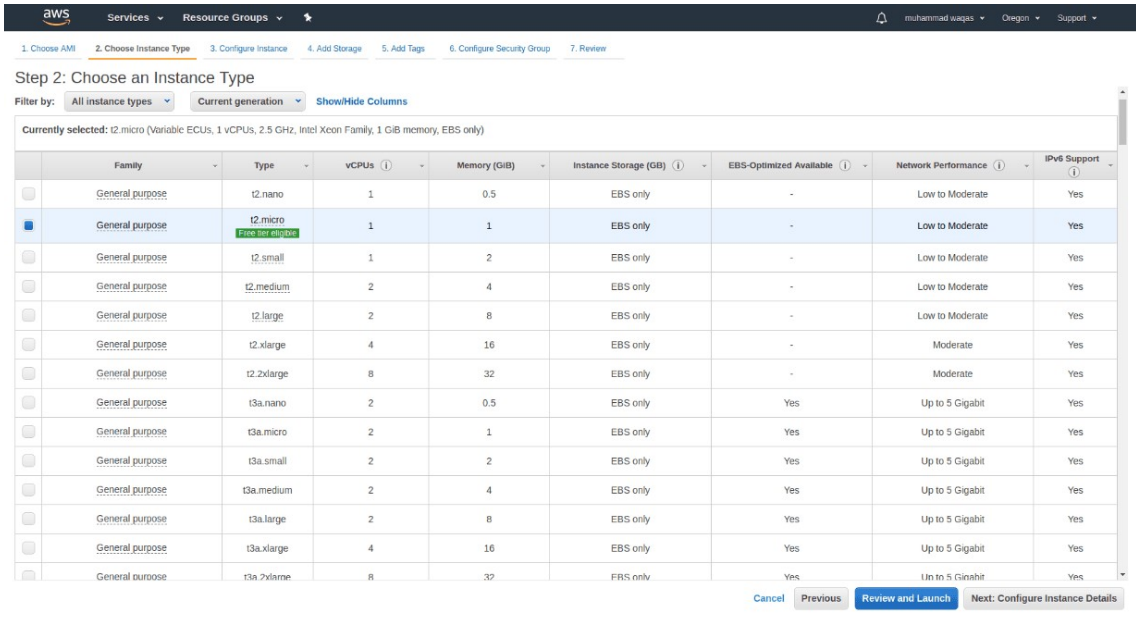
Find free software trial products in the AWS Marketplace from the EC2 Launch Wizard. Or try these popular AMIs:

Barracuda CloudGen Firewall for AWS - PAYG
By Barracuda Networks, Inc.
Rating ★★★★★
Starting from $0.60/hr or from $4,599/yr (12% savings) for software + AWS usage fees
View all Infrastructure Software

Matillion ETL for Amazon Redshift
By Matillion
Rating ★★★★★
Starting from $1.37/hr or from $9,950/yr (17% savings) for software + AWS usage fees
View all Business Software

- **Launch Instance AWS EC2**

  Choose Amazon Machine Image for launching instance. Amazon provides various types of images. However, Ubuntu 18.04 is more secure and 70 servers are using different versions of linux operating system.



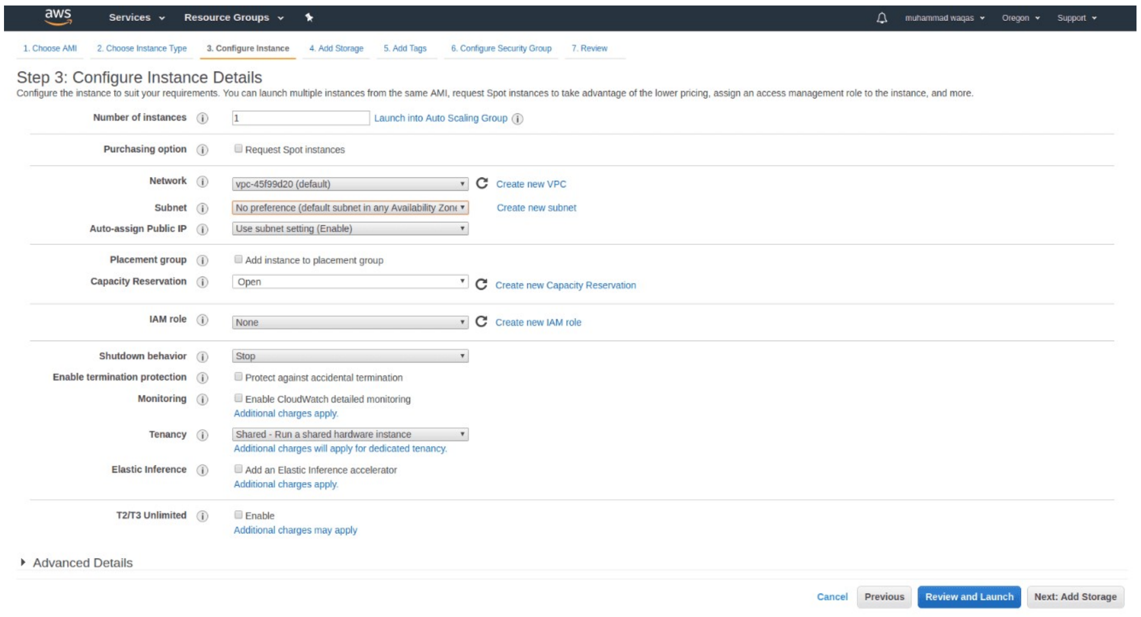
Step 2: Choose an Instance Type

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
☐	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
■	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.xlarge	4	16	EBS only	-	Moderate	Yes
☐	General purpose	t2.2xlarge	8	32	EBS only	-	Moderate	Yes
☐	General purpose	t3a.nano	2	0.5	EBS only	Yes	Up to 5 Gigabit	Yes
☐	General purpose	t3a.micro	2	1	EBS only	Yes	Up to 5 Gigabit	Yes
☐	General purpose	t3a.small	2	2	EBS only	Yes	Up to 5 Gigabit	Yes
☐	General purpose	t3a.medium	2	4	EBS only	Yes	Up to 5 Gigabit	Yes
☐	General purpose	t3a.large	2	8	EBS only	Yes	Up to 5 Gigabit	Yes
☐	General purpose	t3a.xlarge	4	16	EBS only	Yes	Up to 5 Gigabit	Yes
☐	General purpose	t3a.2xlarge	8	32	EBS only	Yes	Up to 5 Gigabit	Yes

- **Amazon AWS EC2 Create instance with Ubuntu 18.04**

  Select Ubuntu 18.04 for image. After selecting the machine, click on “Next Add Storage” button.



Step 3: Configure Instance Details

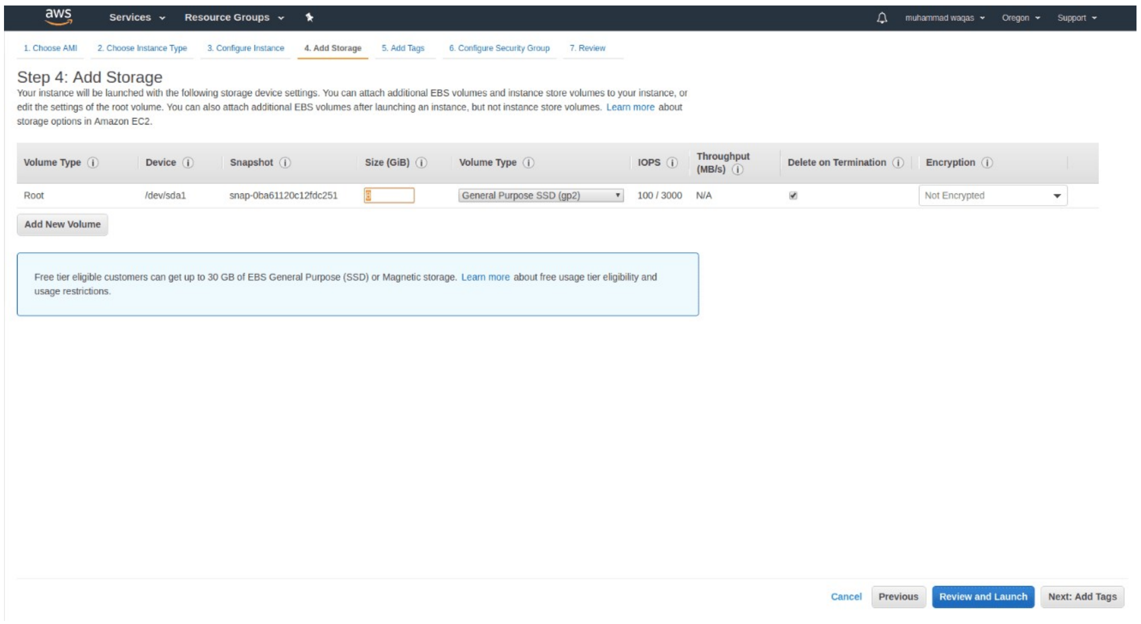
Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances	1 — Launch into Auto Scaling Group
Purchasing option	Request Spot instances
Network	vpc-45f99d20 (default) — Create new VPC
Subnet	No preference (default subnet in any Availability Zone — Create new subnet
Auto-assign Public IP	Use subnet setting (Enable)
Placement group	Add instance to placement group
Capacity Reservation	Open — Create new Capacity Reservation
IAM role	None — Create new IAM role
Shutdown behavior	Stop
Enable termination protection	Protect against accidental termination
Monitoring	Enable CloudWatch detailed monitoring. Additional charges apply.
Tenancy	Shared - Run a shared hardware instance. Additional charges will apply for dedicated tenancy.
Elastic Inference	Add an Elastic Inference accelerator. Additional charges apply.
T2/T3 Unlimited	Enable. Additional charges may apply

Advanced Details

- **AWS Storage**

  Select default storage configuration and click the “Next Add Tags” button.



Step 4: Add Storage

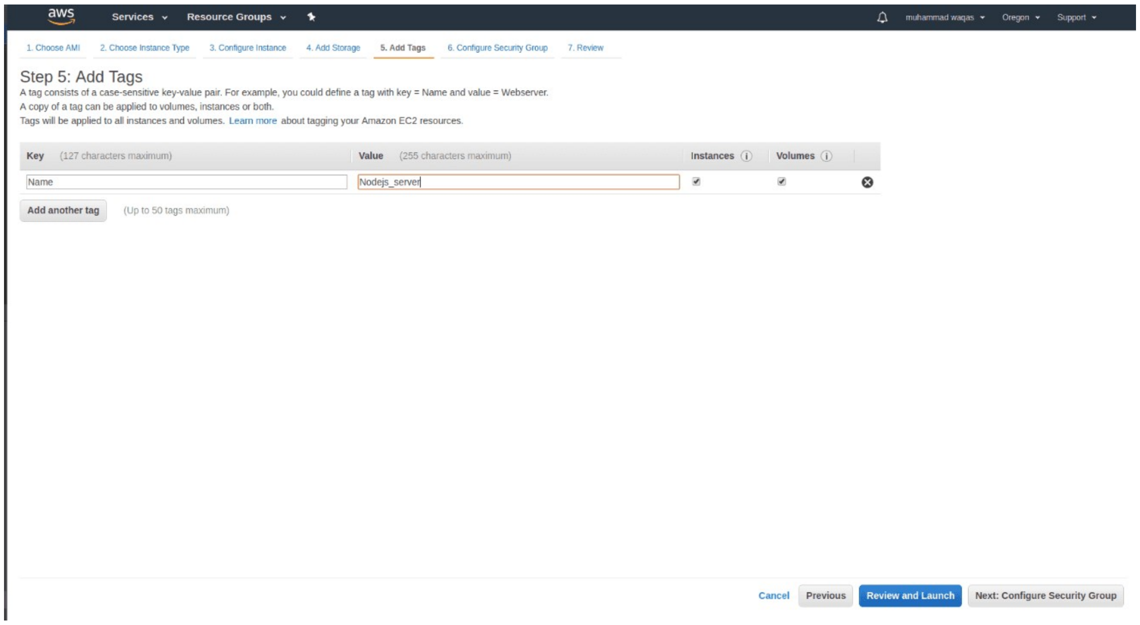
Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encryption
Root	/dev/sda1	snap-0ba61120c12fdc251	8	General Purpose SSD (gp2)	100 / 3000	N/A	✓	Not Encrypted

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.

- **AWS Storage Configuration**

  In Add Tags page section, add at least single tag using key-value pairs. For example, Key=ServerName and value=Nodejs_Server. After adding tag click on Next: Configure Security Group Button.



Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.
A copy of a tag can be applied to volumes, instances or both.
Tags will be applied to all instances and volumes. Learn more about tagging your Amazon EC2 resources.

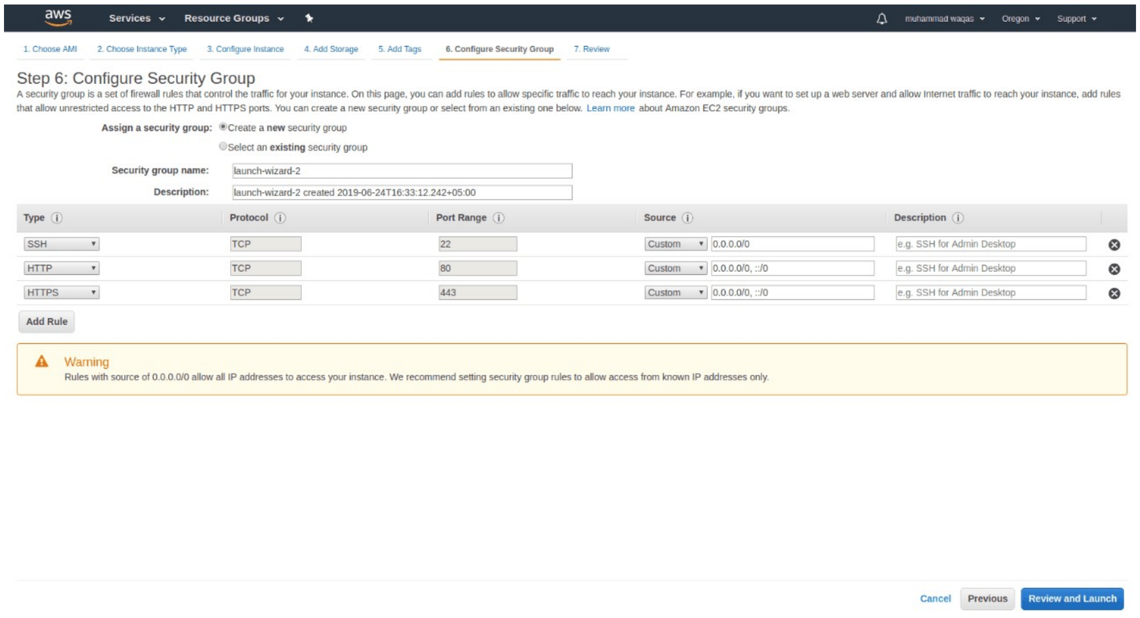
Key (127 characters maximum)	Value (255 characters maximum)	Instances	Volumes
Name	Nodejs_server	✓	✓

Add another tag (Up to 50 tags maximum)

- **AWS EC2 Tags**

After adding these tags, open some ports of server to communicate with outer world. Configure these ports after launching the server. Click on Add Tag button and add 80, 8080, and 22 ports.

Click on Next Review and Launch button.



Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group: [✓ Create a **new** security group] Select an **existing** security group

Security group name: launch-wizard-2
Description: launch-wizard-2 created 2019-06-24T16:33:12.242+05:00

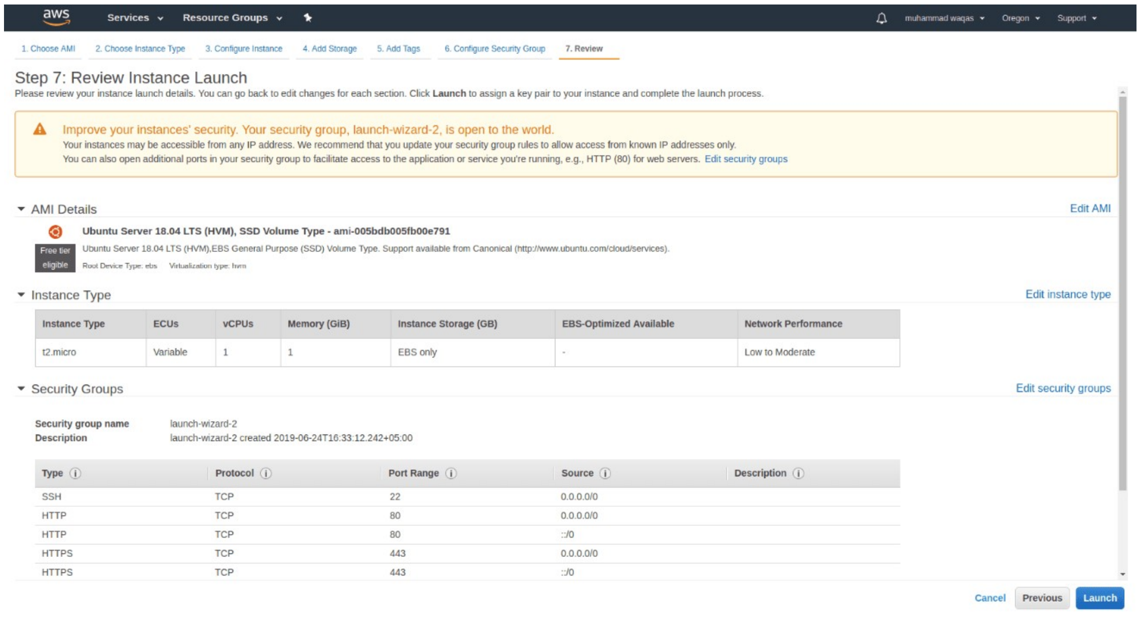
Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Custom 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

Warning
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.

- **Security Groups AWS EC2 Server**

  Click on the “Launch” Button and a dialog box can be seen. Create a new key pair and download it on local machine.

  After generating the key pair click on Launch Instances button.



Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

Improve your instances' security. Your security group, launch-wizard-2, is open to the world.
Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only.
You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. Edit security groups

AMI Details — Edit AMI

Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-005bdb005fb00e791
Free tier eligible
Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Root Device Type: ebs   Virtualization type: hvm

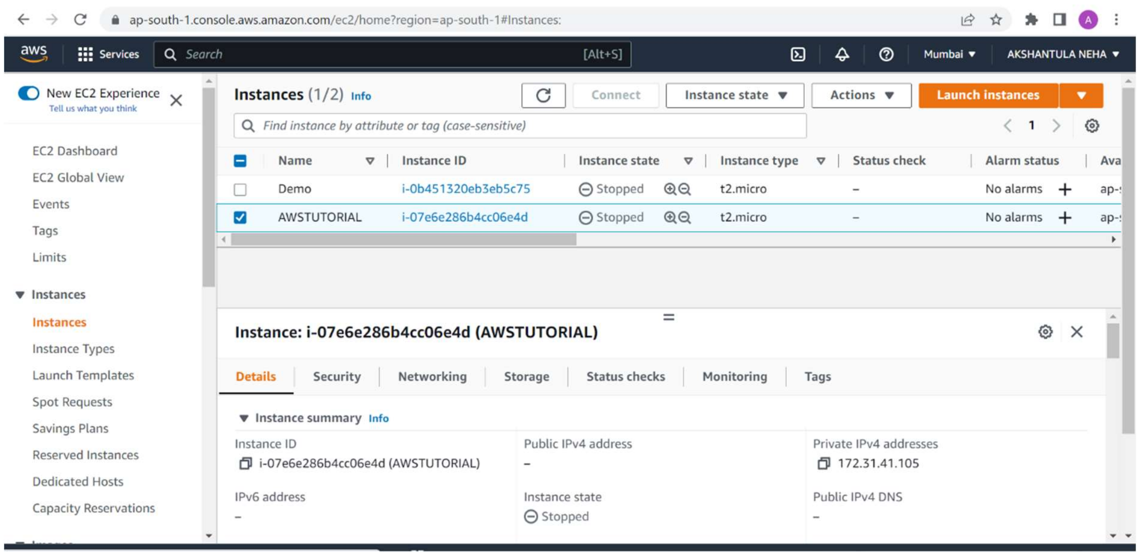
Instance Type — Edit instance type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

Security Groups — Edit security groups

Security group name	launch-wizard-2
Description	launch-wizard-2 created 2019-06-24T16:33:12.242+05:00

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	0.0.0.0/0	
HTTP	TCP	80	0.0.0.0/0	
HTTP	TCP	80	::/0	
HTTPS	TCP	443	0.0.0.0/0	
HTTPS	TCP	443	::/0	



Instances (1/2) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Ava
☐	Demo	i-0b451320eb3eb5c75	Stopped	t2.micro	–	No alarms	ap-
☑	AWSTUTORIAL	i-07e6e286b4cc06e4d	Stopped	t2.micro	–	No alarms	ap-

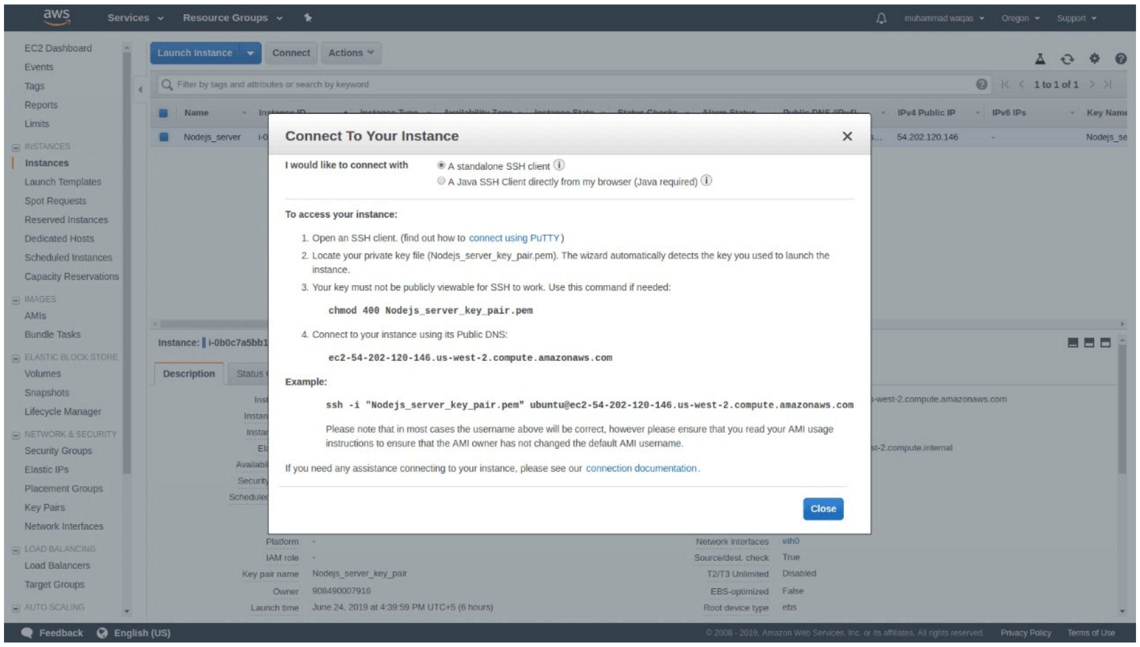
Instance: i-07e6e286b4cc06e4d (AWSTUTORIAL)

Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-07e6e286b4cc06e4d (AWSTUTORIAL)	–	172.31.41.105
IPv6 address	Instance state	Public IPv4 DNS
–	Stopped	–

3. SSH into your instance

Now access the server from local machine using terminal on Linux and putty on Windows. If Windows is used, change .pem key to .ppk format. Go to your local directory where you downloaded the private key. Go to AWS instances console and click on the Connect button.


Connect To Your Instance

I would like to connect with [✓ A standalone SSH client] A Java SSH Client directly from my browser (Java required)

To access your instance:

1. Open an SSH client. (find out how to connect using PuTTY)
2. Locate your private key file (Nodejs_server_key_pair.pem). The wizard automatically detects the key you used to launch the instance.
3. Your key must not be publicly viewable for SSH to work. Use this command if needed:

```
chmod 400 Nodejs_server_key_pair.pem
```

4. Connect to your instance using its Public DNS:

```
ec2-54-202-120-146.us-west-2.compute.amazonaws.com
```

Example:

```
ssh -i "Nodejs_server_key_pair.pem" ubuntu@ec2-54-202-120-146.us-west-2.compute.amazonaws.com
```

Please note that in most cases the username above will be correct, however please ensure that you read your AMI usage instructions to ensure that the AMI owner has not changed the default AMI username.

If you need any assistance connecting to your instance, please see our connection documentation.

Platform	-	Network interfaces	eth0
IAM role	-	Source/dest. check	True
Key pair name	Nodejs_server_key_pair	T2/T3 Unlimited	Disabled
Owner	908490007916	EBS-optimized	False
Launch time	June 24, 2019 at 4:39:59 PM UTC+5 (6 hours)	Root device type	ebs

4. Install Nodejs

After getting connected to the instance, start installing the required packages in the environment for node.js app to work. Install the latest version of Node JS which is 12.x. Once done with the installation type “node -v ” on your terminal, which shows v12.4.0 on terminal screen.

5. Install Git and clone repository from GitHub

- **Install git on Ubuntu**

  There is no need to install git on ubuntu 18.04. It is already installed on server. It can be checked using git --help command. If it is not installed it can be installed using the following command.

  sudo apt-get install git

- **Clone app on server**

  Clone a simple Node js basic app by using this command on terminal.

  git clone https://github.com/johnpapa/node-hello.git

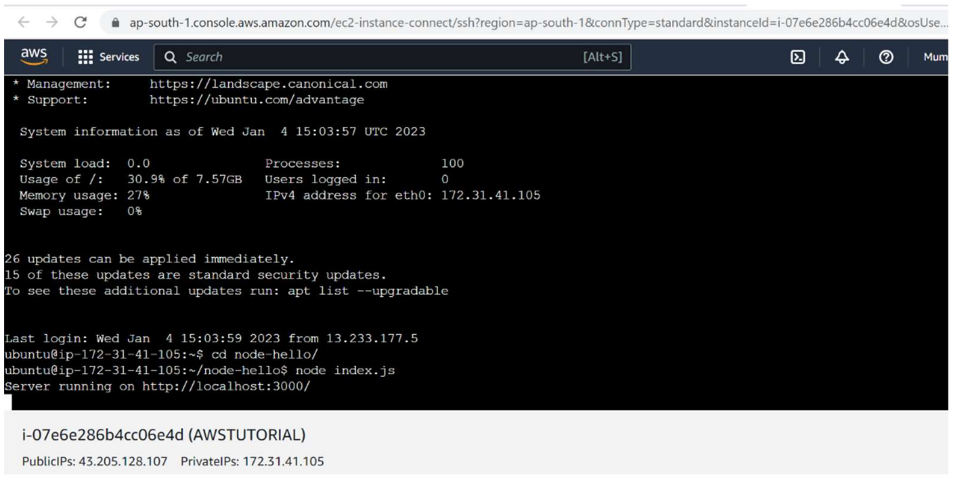
6. Start the node.js app

Run the following command inside our project directory:

npm install

It will install all the required packages for the app.

node index.js



ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-07e6e286b4cc06e4d&osUse...
aws
Services
Search
[Alt+S]
Mum

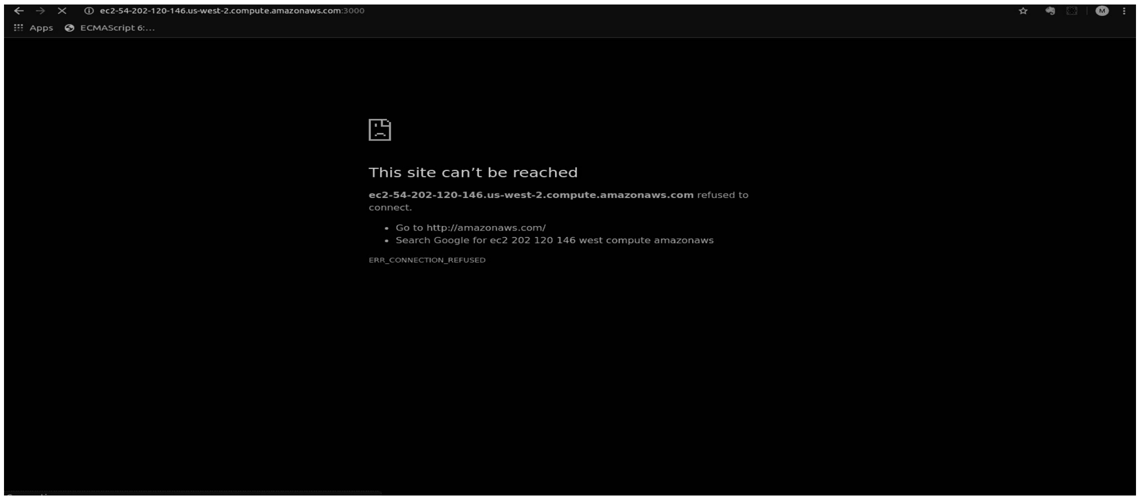
```
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Wed Jan  4 15:03:57 UTC 2023

  System load:  0.0               Processes:             100
  Usage of /:   30.9% of 7.57GB   Users logged in:       0
  Memory usage: 27%               IPv4 address for eth0: 172.31.41.105
  Swap usage:   0%


26 updates can be applied immediately.
15 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable


Last login: Wed Jan  4 15:03:59 2023 from 13.233.177.5
ubuntu@ip-172-31-41-105:~$ cd node-hello/
ubuntu@ip-172-31-41-105:~/node-hello$ node index.js
Server running on http://localhost:3000/
```

i-07e6e286b4cc06e4d (AWSTUTORIAL)

PublicIPs: 43.205.128.107   PrivateIPs: 172.31.41.105

- **AWS EC2 Run Node.js App**

  Copy public DNS from instances page, paste it on web browser, and append port :3000 at the end of the DNS address. This will not work and there is nothing to show on browser and the browser connection will timeout.



This site can’t be reached

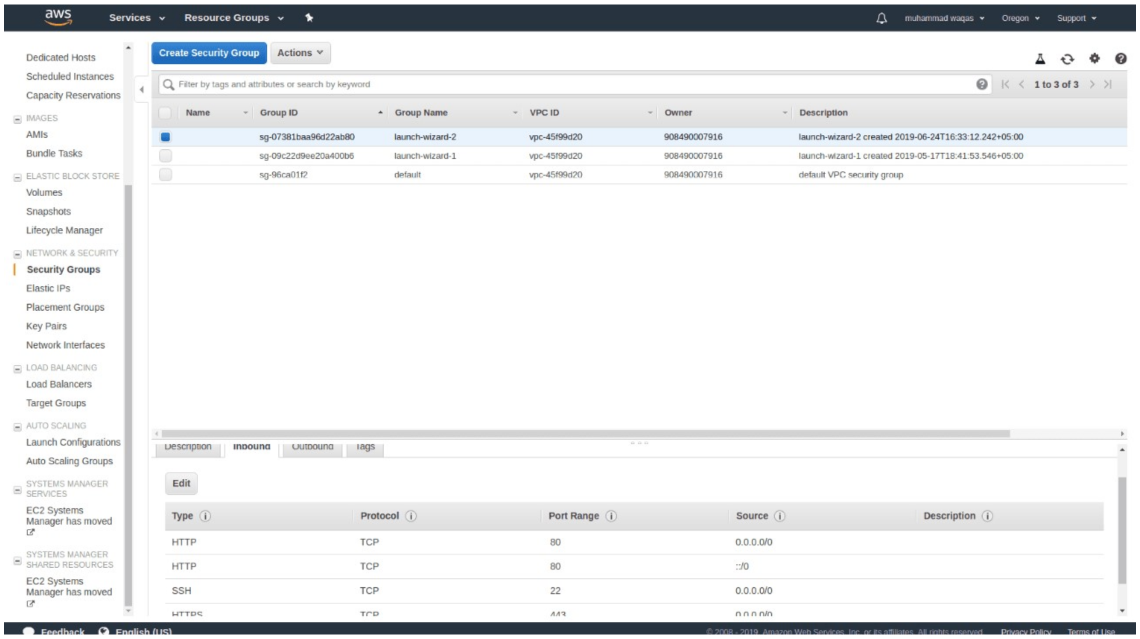
ec2-54-202-120-146.us-west-2.compute.amazonaws.com refused to connect.

- Go to http://amazonaws.com/
- Search Google for ec2 202 120 146 west compute amazonaws

ERR_CONNECTION_REFUSED

- **AWS EC2 Connection Timeout**

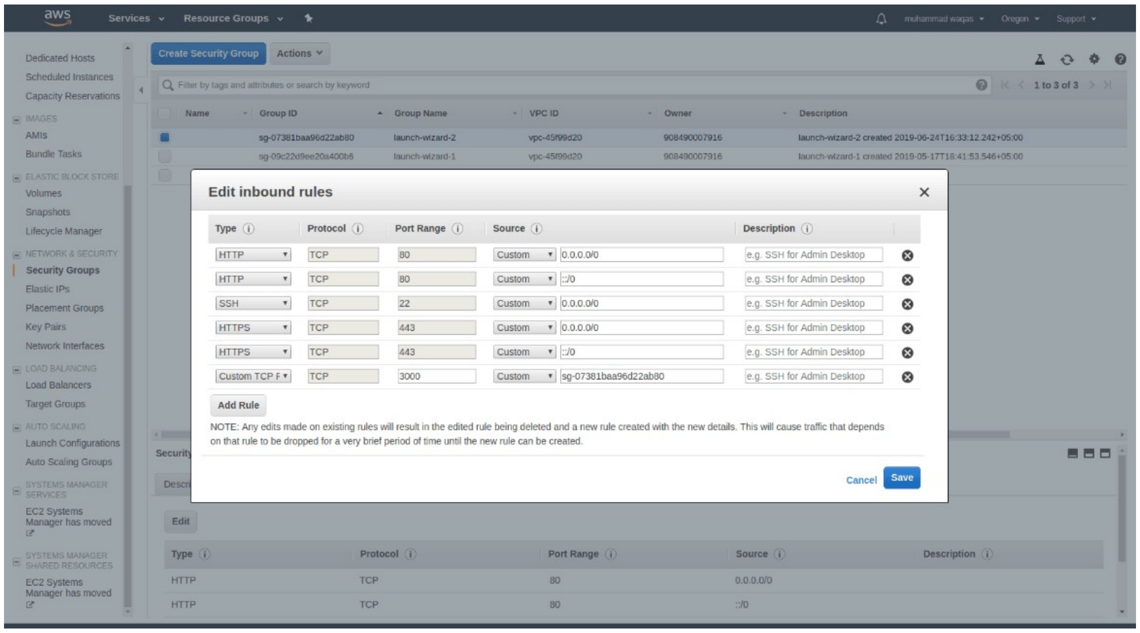
  This is where the Security Group comes in. In the AWS management console, go to Security Groups, select the one not named 'default' (launch-wizard-1 or your security group name) and edit the Inbound rules.



Name	Group ID	Group Name	VPC ID	Owner	Description
	sg-07381baa96d22ab80	launch-wizard-2	vpc-45f99d20	908490007916	launch-wizard-2 created 2019-06-24T16:33:12.242+05:00
	sg-09c22d9ee20a400b6	launch-wizard-1	vpc-45f99d20	908490007916	launch-wizard-1 created 2019-05-17T18:41:53.546+05:00
	sg-96ca01f2	default	vpc-45f99d20	908490007916	default VPC security group

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	0.0.0.0/0	
HTTP	TCP	80	::/0	
SSH	TCP	22	0.0.0.0/0	
HTTPS	TCP	443	0.0.0.0/0	

© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Edit inbound rules

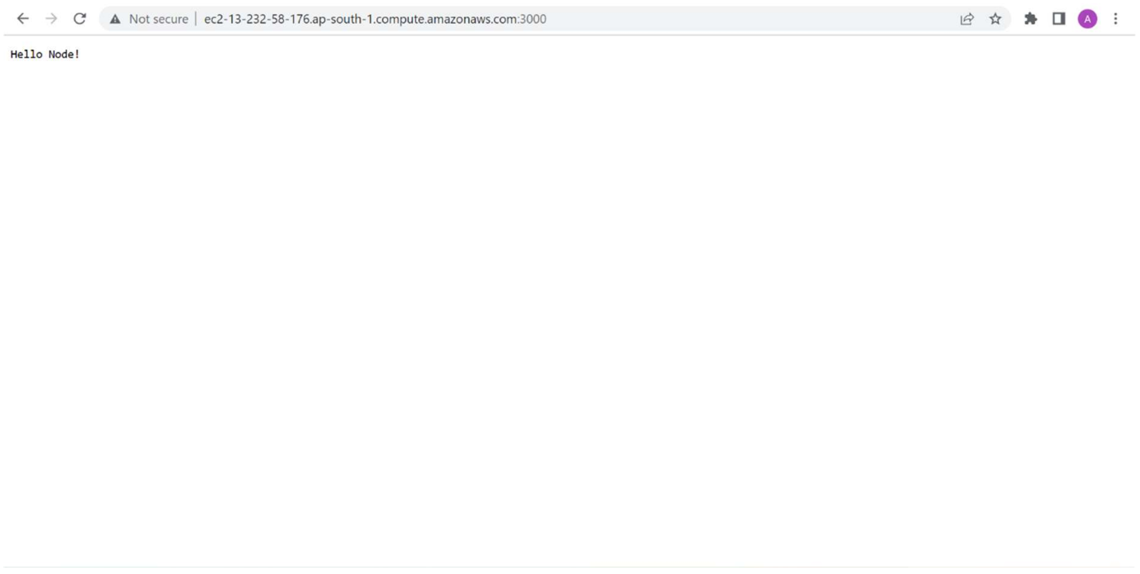
Type	Protocol	Port Range	Source		Description
HTTP	TCP	80	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom	::/0	e.g. SSH for Admin Desktop
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Custom	::/0	e.g. SSH for Admin Desktop
Custom TCP F	TCP	3000	Custom	sg-07381baa96d22ab80	e.g. SSH for Admin Desktop

NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.

- **AWS EC2 Inbound Rules**

  Access the URL again and the app content can be seen now.

Now even if you close the terminal and check the url in the browser, the app will be running.


Not secure | ec2-13-232-58-176.ap-south-1.compute.amazonaws.com:3000

```
Hello Node!
```

Chapter 3

Conclusion

AWS cloud service has a wide range of utilities listed as follows that help to deploy web applications on cloud:

i. Easy to Use

AWS's platform is clearly expressed and even a neophyte can use it. There won't be any problem for a new applicant as well as for an existing applicant. This is possible due to the AWS Management Console or well-documented web services.

ii. No Capacity Limits

Organizations launch different projects and the guess what capacity they will need. AWS helps them by providing this capacity at a minimum cost. Through this benefit, their workload is decreased and they can focus and built different ideas. The customers predict the capacity and they pay higher prices than that but AWS provide them capacity at low-cost. The moment you feel like you should increase your capacity you can do it freely. Moreover, if you realize that you are not in need of so much storage you can get back to the previous storage and all you have to pay for what you use.

iii. Provides Speed and Agility

In the old world if we talk to an engineer, Enterpriser or a company about how long will it take to hire a server, the answer we will get is 1 week. But AWS provides us within minutes. All you have to do is select your requirement and you can proceed without talking to anyone as it is easy and flexible.

With this, you can quickly deploy your application. AWS provides us with tools which helps us to reduce the time we spend on a task such as Auto Scaling, AWS Tools and Elastic Load Balancing you can select them n the basis of your demand. These applications can be accessed any time you need them.

iv. Secure and Reliable

Amazon allows you to innovate and scale while keeping a secure environment and all you have to pay only for the services you use. AWS provides an end-to-end approach which secures and hardens your infrastructure. Amazon Web Service provide you with the security you need at a lower cost than in an on-premises environment. AWS provides security and also helps to protect the privacy as it is stored in AWS data centres. AWS infrastructure is designed to keep your data safe no matter what size of your data is. It just scales with your AWS cloud usage. AWS manages the highest standard of security and this is the reason users rely on AWS.

Chapter 4

References

[1] https://tkssharma-devops.gitbook.io/devops-training/assignments/assignment-03

[2] https://learnubuntu.com/install-node/

[3] https://www.digitalocean.com/community/tutorials/how-to-install-node-js-on-ubuntu-20-04

[4] https://data-flair.training/blogs/aws-advantages/