**CHAPTER-1**

**AWS CLOUD**

**1.1 Introduction**

Now days, E-commerce website are very popular. Online shopping Websites plays a major part some of them are Flipkart, Amazon etc,These websites are requesting (or) accessing will be varied based on the timings. For suppose Monday to Friday will get approximately 10,000 requests. On weekends will get 50,000 requests and also on festivals, offers and also big billion day, in these timings will get lakhs of lakhs request. For this website, we have to maintain servers, applications, storage, etc. On weekends load will be normally so it may require to manage 2 or 3 servers. On weekends we require 10 to 100 servers. On offer’s day we have to maintain thousands of servers. Again, we have to buy more servers compare to all others days infrastructure. In normal days we don’t require this much of servers, applications,storage. This much of infra of websites building and managing,

it is very difficult.

To overcome this issue, we are moving to cloud. That is AWS Cloud, it is providing a feature called load balancing and Auto scaling. Whatever load is coming to websites, that entire load will be redistributed to all existed servers. When the load is exhausted, to up and run 24x7. AWS increases the servers count, based on the policy. In that policy we have to mention C.P.U percentage. For suppose C.P.U percentage is more than 80% then create 3 more servers and when C.P.U percentage is less than 40% delete the new created servers.

Day by day data is increasing like anything not able to manage with storage clusters like an NFS, SAN, and RAID’S. AWS Database Migration Service helps you migrate databases to AWS quickly and securely.

**1.2 Requirements**

Requirement analysis is a software engineering task that bridges the gap between system level software allocation and software design. It provides system engineer to specify software function and performance indicate software’s interface with the other system elements and establish constraints that software must need.

**1.2.1 Hardware Requirements:**

Processor: t2.micro

RAM:2GB

HDD:8GB

**1.2.2 Software Requirements:**

Operating System: Amazon Linux

Application server: Apache

Front end: HTML, CSS, Bootstrap

Backend: Nodejs, MySQL

Cisco Networking Devices (Switch and Router)

**1.2.3 Services and Platforms:**

AWS (VPC, S3,Linux,SSH,Cloud Watch)

Load Balancing

Auto Scaling

**CHAPTER-2**

**Existing System**

When the user hits on a certain URL and if the requests are more then there will be increase in the traffic load. There will be lagging of the site and can’t be accessed and there will be unavailability of the URL for the users. If the load on the webpage is more on Big Billion Days for a shopping cart site, then huge loss to the Commercial Applications. Purchasing of physical servers only for one day sale is waste of money and configuring every server day by day is a difficult task to the administrator. Maintenance of servers is difficult. This much of infra of websites building and managing, it is challenging.

**CHAPTER-3**

**PROPOSED SYSTEM**

To overcome this issue, we are moving to the cloud. That is AWS CLOUD, and it is providing a feature called load balancing and Auto scaling. So when the load hits on the websites, then automatically virtual servers are created so that entire load will be redistributed to all servers. When the load is exhausted, to up and run 24x7. AWS increases the servers count, based on the policy.

**CHAPTER-4**

**SYSTEM DESIGN**

**4.1 Introduction**

The architecture or the structure of our project is explained to have a brief idea of how it gets processed to meet the requirements of the user.

**4.2** **UML Diagrams**:

**4.2.1 Use Case Diagram:**

Use case Diagrams represent the functionality of the system from a user’s point of view. Use case focuses on the behaviour of the system from an external point of view.

1. **Use Case:**

Use case describes the behaviour of a system. It is used to structure things in a model. It contains multiple scenarios, each of which describes a sequence of actions that is clear enough for outsiders to understand.

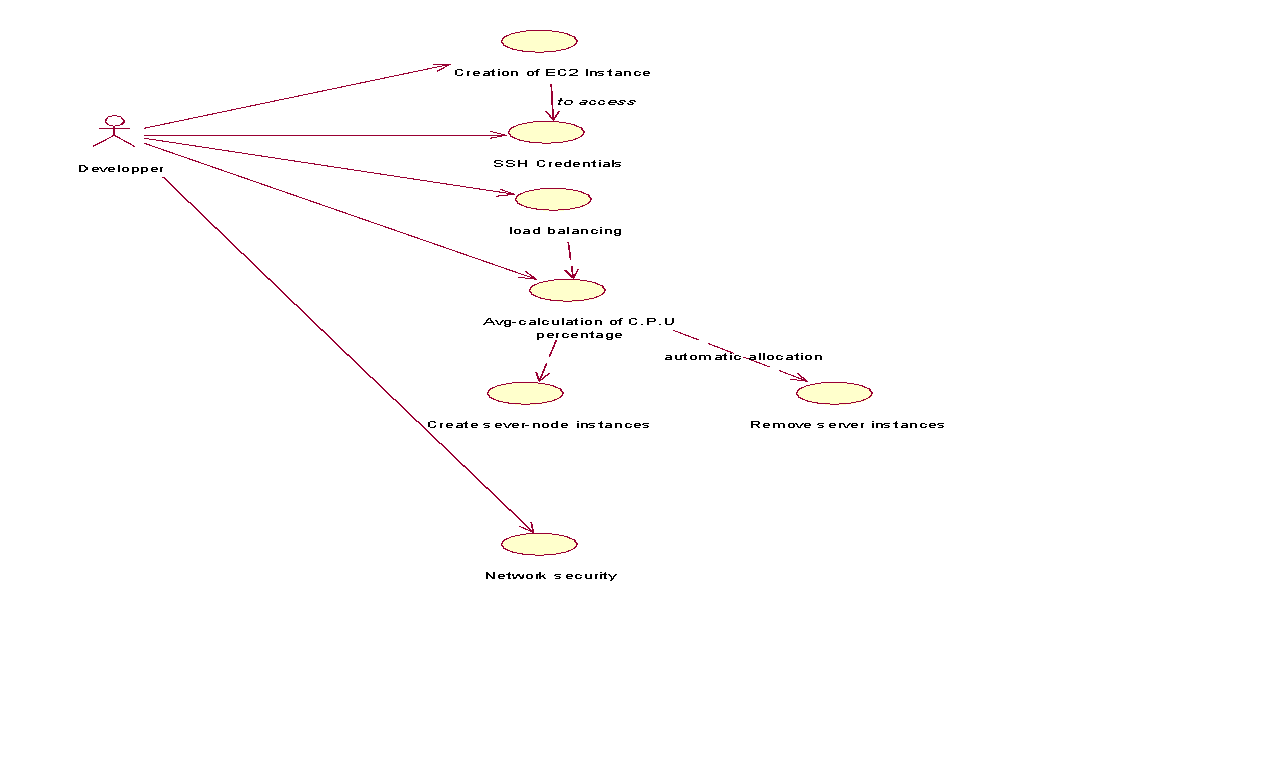
1. **Actor:**

An actor represents a coherent set of roles that users of a system play when interacting with the use cases of the system. An actor participates in use cases to accomplish an overall purpose. An actor can represent the role of a human, a device, or any other system.

1. **Description:**

When a commercial client comes to have a deal with an idea for providing the advance maintenance of his data and security policies. He can choose AWS as the better platform. The developer creates web servers to run the commercial website, provides secure login credentials to the user. If the number of users increases beyond the range then load balancing and auto-scaling comes up, the average percentage of c.p.u is calculated and allocates the web instances and remove the instances when the percentage goes down.

**Use Case View**

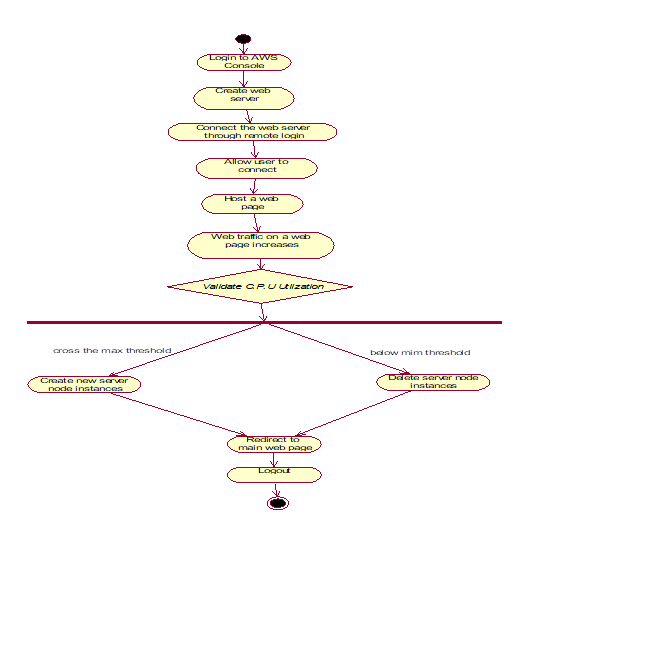


**Fig 4.1: Use case view**

**4.2.2Activity Diagram:**

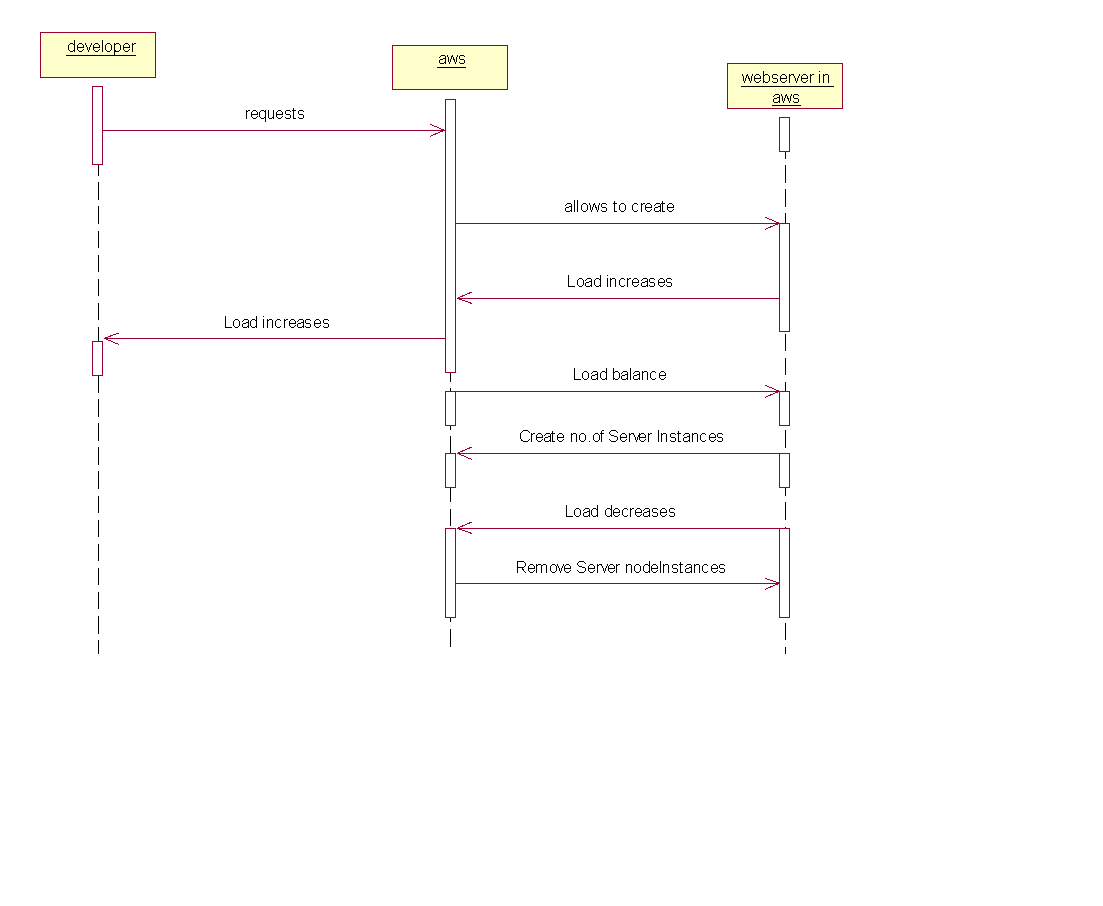
Activity diagram illustrates the dynamic nature of a system by modelling the flow of control from activity. An activity represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow and internal operation.

Action state represents the no interruptible actions of objects. Action flow represents arrows illustrate the relationships among action states.

**Fig 1.1:** Scale In and Out Instances

**4.2.3Sequence Diagram:**

**S**equence diagram describes the pattern of communication among set of interacting objects. An object interacts with other objects by sending messages. The reception of a message by an object triggers the execution of an operation, which in many turns may send messages to other objects.

**Fig 1.3: AWS working**

**CHAPTER-5**

**AWS CLOUD**

**Amazon Web Services (AWS):** is owned by Amazon that provides On-demand cloud computing platforms to individuals, companies, and governments, on a paid subscription basis. The technology allows subscribers to have at their disposal a virtual cluster of computers, available at all time through the Internet.

**5.1 Why AWS?**

One of the most successful business cloud platforms is amazon web services, it mainly focus on infrastructure as a service. No other cloud vendor will not defeat Aws cloud. Because it is the first invented, what are all the strategy and concept of technology of Aws, Microsoft azure is doing copy and paste. Aws having highest revenue compare to all other clouds available in market of world. One of most wonderful features is, it supports all programming languages. Those are Dotnet, C#, Python, Shell Scripting, Go. Etc.

Developing, managing, and operating your applications requires a wide variety of technology services. Customers often ask us what represents a fully-functional, flexible technology infrastructure platform. Below, we outline requirements for a modern, robust, industry-leading technology infrastructure platform with all the benefits that the cloud brings to bear. We also provide information about how AWS delivers against these requirements and why you might need each of these capabilities.

AWS began offering its technology infrastructure platform in 2006. At this point, we have over a million active customers using AWS in every imaginable way, and have developed considerable experience operating at scale. We’ve also innovated and delivered at a very rapid pace (delivering 159 significant features and services in 2012, 280 in 2013, 516 in 2014, 722 in 2015, and 1,017 in 2016). Expect this focus on rapidly delivering what customers want to continue.

**5.2 What is Cloud Computing?**

Cloud computing is the on-demand delivery of compute power, database storage ,applications, and other IT resources through a cloud services platform via the internet with pay-as-you-go pricing.

**Cloud Computing Basics**

Whether you are running applications that share photos to millions of mobile users or you’re supporting the critical operations of your business, a cloud services platform provides rapid access to flexible and low-cost IT resources. With cloud computing, you don’t need to make large upfront investments in hardware and spend a lot of time on the heavy lifting of managing that hardware. Instead, you can provision exactly the right type and size of computing resources you need to power your newest bright idea or operate your IT department. You can access as many resources as you need, almost instantly, and only pay for what you use.

**How Does Cloud Computing Work?**

Cloud computing provides a simple way to access servers, storage, databases and a broad set of application services over the Internet. A Cloud services platform such as Amazon Web Services owns and maintains the network-connected hardware required for these application services, while you provision and use what you need via a web application.

* 1. **Cloud computing Type:**

1. Infrastructure as a service

2. Platform as a service

3. Software as a service

5.3.1 **Infrastructure as a service (IAAS):**

Infrastructure as a service (IAAS) is a form of cloud computing that provides virtualized computing resources over the internet. IAAS is one of the three main categories of cloud computing services, alongside software as a service (SAAS) and platform as a service (PAAS).

**IAAS architecture and how it works:**

The IAAS provider also supplies a range of services to accompany those infrastructure components. These can include detailed billing, monitoring, log access, security, load balancing and clustering, as well as storage resiliency, such as backup, replication and recovery. These services are increasingly policy-driven, enabling IAAS users to implement greater levels of automation and orchestration for important infrastructure tasks. For example, a user can implement policies to drive load balancing to maintain application availability and performance.

IAAS customers access resources and services through a wide area network (WAN), such as the internet, and can use the cloud provider's services to install the remaining elements of an application stack. For example, the user can log in to the IAAS platform to create virtual machines (VMs); install operating systems in each VM; deploy middleware, such as databases; create storage buckets for workloads and backups; and install the enterprise workload into that VM. Customers can then use the provider's services to track costs, monitor performance, balance network traffic, troubleshoot application issues, and manage disaster recovery and more. Any cloud computing model requires the participation of a provider. The provider is often a third-party organization that specializes in selling IAAS. Amazon Web Services (AWS) and Google Cloud Platform (GCP) are examples of independent IAAS providers. A business might also opt to deploy a private cloud, becoming its own provider of infrastructure services. Organizations choose IAAS because it is often easier, faster and more cost-efficient to operate a workload without having to buy, manage and support the underlying infrastructure. With IAAS, a business can simply rent or lease that infrastructure from another business.

IAAS is an effective model for workloads that are temporary, experimental or that change unexpectedly. For example, if a business is developing a new software product, it might be more cost-effective to host and test the application using an IAAS provider. Once the new software is tested and refined, the business can remove it from the IAAS environment for a more traditional, in-house deployment. Conversely, the business could commit that piece of software to a long-term IAAS deployment, where the costs of a long-term commitment may be less.

In general, IAAS customers pay on a per use basis, typically by the hour, week or month. Some IAAS providers also charge customers based on the amount of virtual machine space they use. This pay-as-you-go model eliminates the capital expense of deploying in-house hardware and software.

When a business cannot use third-party providers, a private cloud built on premises can still offer the control and scalability of IAAS -- though the cost benefits no longer apply.

5.3.2 **Platform as a service (PAAS):**

Platform as a service (PAAS) is a cloud computing model in which a third-party provider delivers hardware and software tools -- usually those needed for application development -- to users over the internet. A PAAS provider hosts the hardware and software on its own infrastructure. As a result, PAAS frees users from having to install in-house hardware and software to develop or run a new application.

**PAAS architecture and how it works**

PAAS does not typically replace a business's entire IT infrastructure. Instead, a business relies on PAAS providers for key services, such as application hosting or Java development.A PAAS provider builds and supplies a resilient and optimized environment on which users can install applications and data sets. Users can focus on creating and running applications rather than constructing and maintaining the underlying infrastructure and services.Many PAAS products are geared toward software development. These platforms offer compute and storage infrastructure, as well as text editing, version management, compiling and testing services that help developers create new software more quickly andefficiently. A PAAS product can also enable development teams to collaborate and work together, regardless of their physical location.

5.3.3 **Software as a service (SAAS):**

Software as a service (SAAS) is a software distribution model in which a third-party provider hosts application and makes them available to customers over the Internet. SAAS is one of three main categories of cloud computing, alongside infrastructure as a service (IAAS) and platform as a service (PAAS).

SAAS is closely related to the application service provider (ASP) and on demand computing software delivery models. The hosted application management model of SAAS is similar to ASP, where the provider hosts the customer’s software and delivers it to approved end users over the internet. In the software on demand SAAS model, the provider gives customers network-based access to a single copy of an application that the provider created specifically for SAAS distribution. The application’s source code is the same for all customers and when new features or functionalities are rolled out, they are rolled out to all customers. Depending upon the service level agreement (SLA), the customer’s data for each model may be stored locally, in the cloud or both locally and in the cloud. Organizations can integrate SAAS applications with other software using application programming interfaces (APIs). For example, a business can write its own software tools and use the SAAS provider's APIs to integrate those tools with the SAAS offering.

There are SAAS applications for fundamental business technologies, such as email, sales management, customer relationship management (CRM), financial management, human resource management (HRM), billing and collaboration. Leading SAAS providers include Salesforce, Oracle, SAP, Intuit and Microsoft SAAS applications are used by a range of IT professionals and business users, as well as C-level executives.

* 1. **Advantages and Benefits of Cloud Computing**

**1.Trade capital expense for variable expense:** Instead of having to invest heavily in data centres and servers before you know how you’re going to use them, you can only pay when you consume computing resources, and only pay for how much you consume.

**2. Benefit from massive economies of scale**

By using cloud computing, you can achieve a lower variable cost than you can get on your own. Because usage from hundreds of thousands of customers are aggregated in the cloud, providers such as Amazon Web Services can achieve higher economies of scale which translates into lower pay as you go prices.

**3. Stop guessing capacity**

Eliminate guessing on your infrastructure capacity needs. When you make a capacity decision prior to deploying an application, you often either end up sitting on expensive idle resources or dealing with limited capacity. With cloud computing, these problems go away. You can access as much or as little as you need, and scale up and down as required with only a few minutes’ notice.

**4. Increase speed and agility**

In a cloud computing environment, new IT resources are only ever a click away, which means you reduce the time it takes to make those resources available to your developers from weeks to just minutes. This results in a dramatic increase in agility for the organization, since the cost and time it takes to experiment and develop is significantly lower.

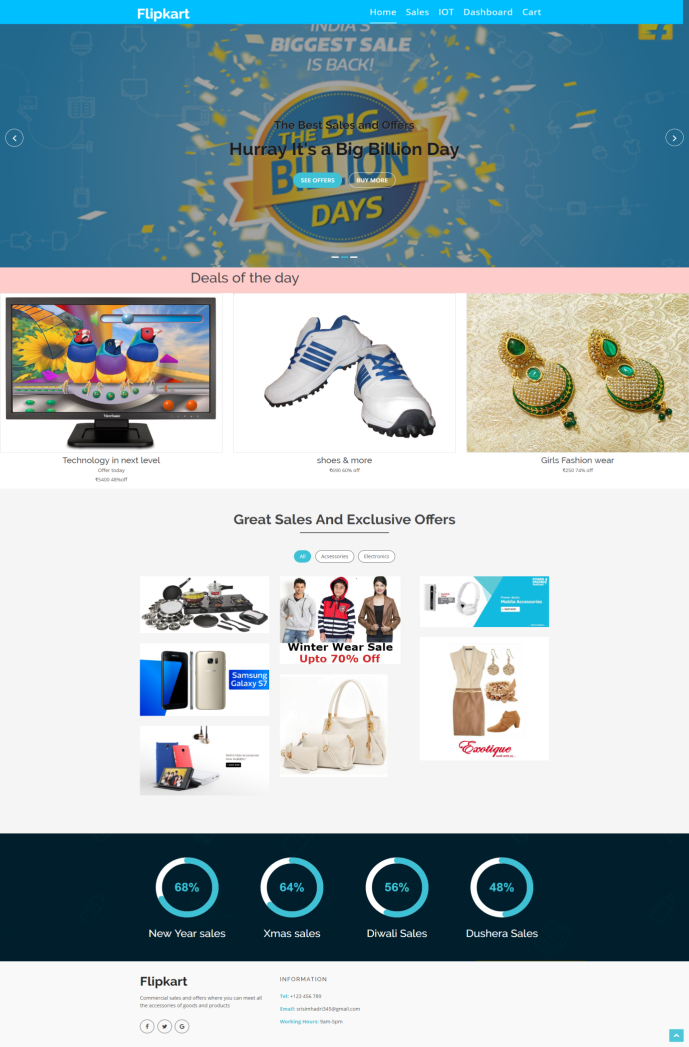
1. **Stop spending money on running and maintaining data centres**

Focus on projects that differentiate your business, not the infrastructure. Cloud computing lets you focus on your own customers, rather than on the heavy lifting of racking, stacking and powering servers.

**CHAPTER-6**

**WEBSITE AND DATABASE DETAILS**

To explain these auto-scaling and load balancing functionalities on a cloud, we have designed a web portal using HTML, CSS, JavaScript, and Bootstrap as the programming languages. Index.html is the main web page that was designed to launch on to aEC2 instance.



**Fig 6.1: Main website**

**CHAPTER-**

**IOT INTRODUCTION**

IOT means Internet of Things, the main aim of our project AWS Cloud and Network Security is to retrieve the data of cloud from anywhere at any time, if we have Internet Connection. In this project we are using sensors for getting information about temperature, humidity, rainfall at a particular location.

In this process, we have used different types of sensors they are For temperature and humidity information we are using DHT11 sensor. This is connected with Nodemcu.

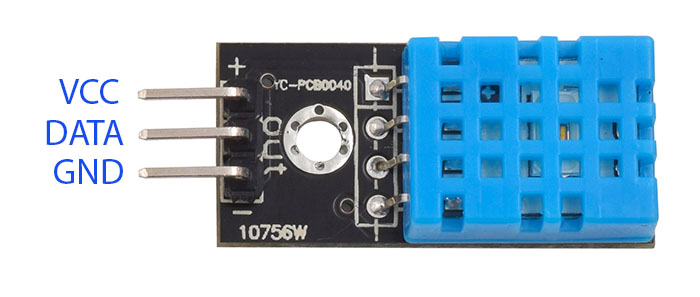
**6.1.1 DHT11 temperature and Humidity sensor**

DHT11 sensor is used for measuring both humidity and temperature values. It can measure relative humidity in percentage (20 to 90% RH) and temperature in degree Celsius in the range of 0 to 50°C.It has 3 pins. They are:

1. VCC

2. DATA OUT

3. GND



**Fig 6.2: DHT 11**

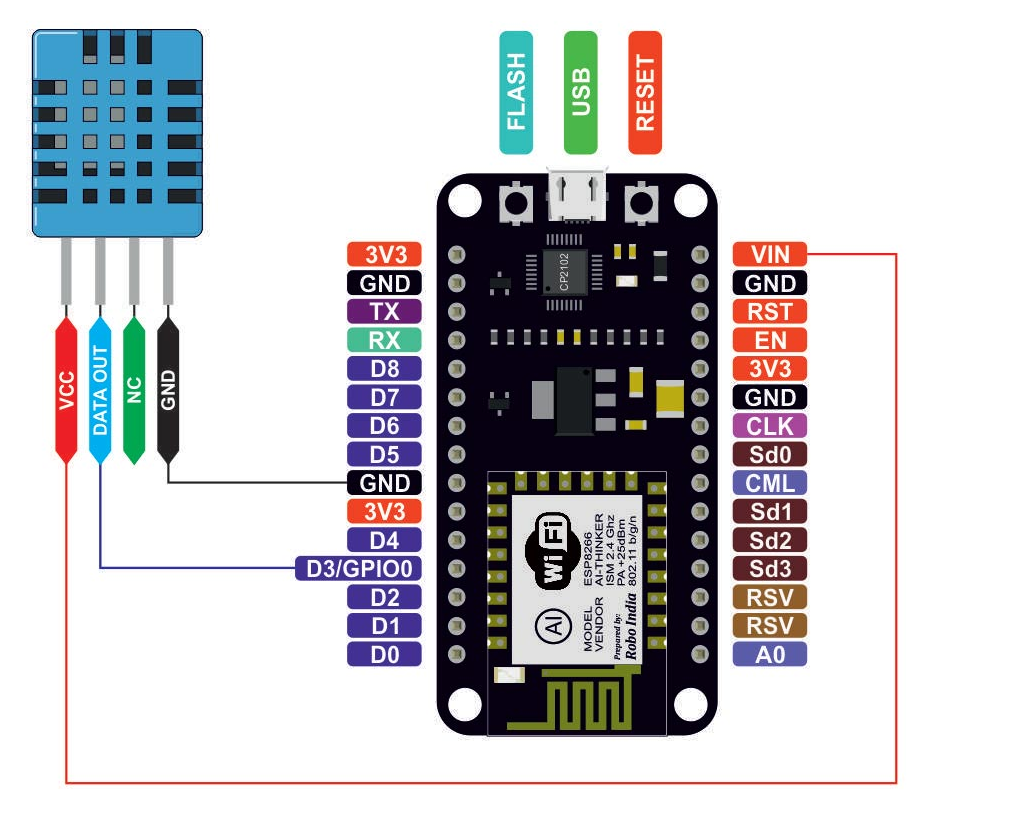
# Nodemcu:



**Fig 6.3: Node MCU**

NodeMCU Dev. Kit/board consist of ESP8266 Wi-Fi enabled chip. The **ESP8266** is a low-cost [Wi-Fi](https://en.wikipedia.org/wiki/Wi-Fi) chip developed by Espressif Systems with TCP/IP protocol.NodeMCU Dev Kit has **Arduino like** Analog (i.e. A0) and Digital (D0-D8) pins on its board. It supports serial communication protocols i.e. UART, SPI, I2C etc. Using such serial protocols we can connect it with serial devices like I2C enabled LCD display, Magnetometer HMC5883, MPU-6050 Gyro meter + Accelerometer, RTC chips, GPS modules, touch screen displays, SD cards etc.

# 6.1.2 DHT11 SENSOR CONNECTION WITH NODEMCU:



**Fig 6.4: Pin Diagram of Node MCU**

**Connection of Nodemcu With Dht11 Sensor:**  
**Pin 1** of the DHT11 goes into **+3v** of the NodeMCU.

**Pin 2** of the DHT11 goes into Digital Pin **D3** of the NodeMCU.

**Pin 3** of the DHT11 goes into Ground Pin (**GND**) of the NodeMCU.

**6.1.3** **Rain Drop Sensor**

The rain sensor module is an easy tool for rain detection. It can be used as a switch when raindrop falls through the raining board and also for measuring rainfall intensity. The module features, a rain board and the control board that is separate for more convenience, power indicator LED and an adjustable sensitivity though a potentiometer

The analog output is used in detection of drops in the amount of rainfall. Connected to 5V power supply, the LED will turn on when induction board has no rain drop, and DO output is high. When dropping a little amount water, DO output is low, the switch indicator will turn on. Brush off the water droplets, and when restored to the initial state, outputs high level.

**Specifications**

• Adopts high quality of RF-04 double sided material.

• Area: 5cm x 4cm nickel plate on side.

• Anti-oxidation, anti-conductivity, with long use time.

• Comparator output signal clean waveform is good, driving ability, over 15mA.

• Potentiometer adjust the sensitivity.

• Working voltage 5V.

• Output format: Digital switching output (0 and 1) and analog voltage output AO.

• With bolt holes for easy installation.

• Small board PCB size: 3.2cm x 1.4cm.

• Uses a wide voltage LM393 comparator.

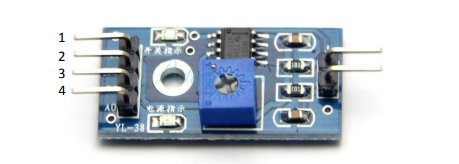
Pin Configuration

1. VCC: 5V DC

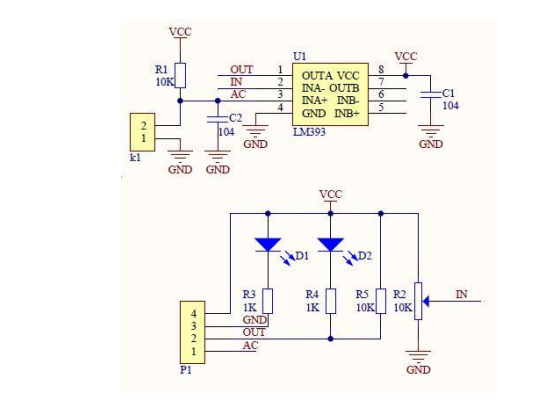
2. GND: ground

3. DO: high/low output

4. AO: analog output 1 2 3 4

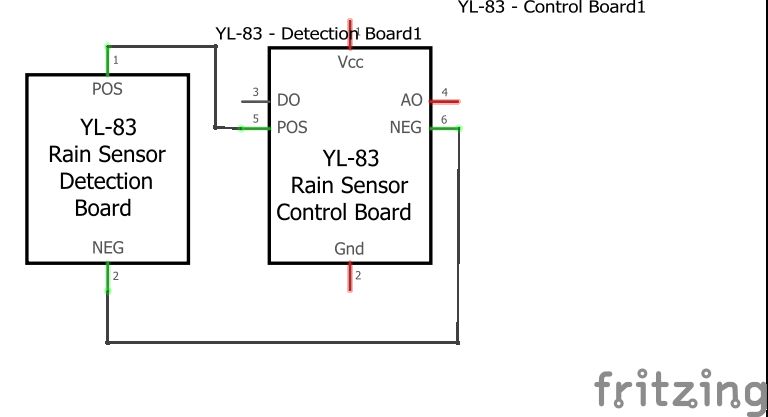


**Schematic Diagram**



**Fig 6.5:** **Schematic Diagram**

**Pin configuration of get spares rain drop sensor:**

****

**Fig 6.6:** **Pin Configuration of Raining Sensor**

**Connecting sensor on bread board:**

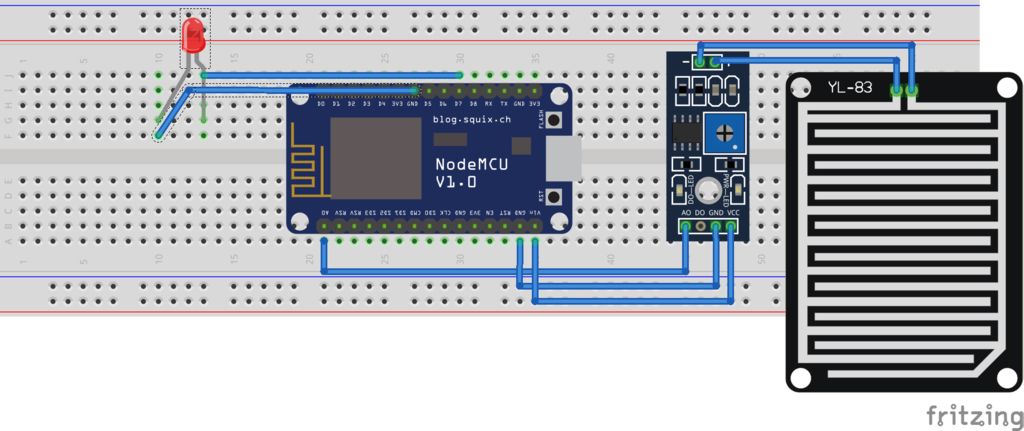


Fig 6.7: Connecting of Rain Sensor with Node MCU

**Working Principle of Raindrop Sensor**

[](https://cdn.instructables.com/FZF/E2NC/JK2UIKLW/FZFE2NCJK2UIKLW.LARGE.jpg)

Raindrop sensor is basically a board on which nickel is coated in the form of lines. It works on the principal of resistance. When there is no rain drop on board. Resistance is high so we get high voltage according to V=IR. When rain drop present it reduces the resistance because water is conductor of electricity and presence of water connects nickel lines in parallel so reduced resistance and reduced voltage drop across it.

**6.2 Installation of libraries:**

You need to install the **DHTLib** library. It has all the functions needed to get the humidity and temperature readings from the sensor. It’s easy to install.

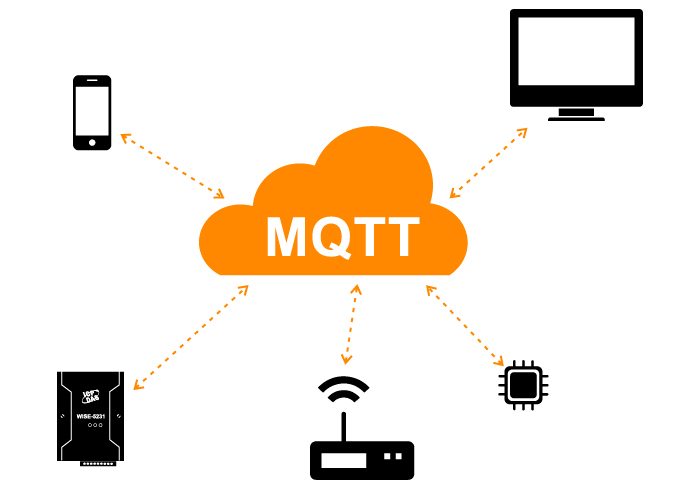
Open up the **Arduino IDE,** then go to **Sketch** > **Include Library** > **Manage Libraries** > **Search DHTLib**

After it’s installed, upload this program to the Node MCU and check output in the serial monitor.

1. After connecting the devices, the code is written and send to the cloud through MQTT protocol.
2. There the code is uploaded into Arduino and it shows live data of temperature and humidity of a particular location for every 5 minutes.

**6.3 MQTT PROTOCOL:**

MQTT means Message Queuing Telemetry Transport. It is small size, light weight, low power usage, minimized data packets and ease of implementation .it is used for “machine-to-machine” connection.



**Fig 6.8: MQTT Protocol**

**Features:**

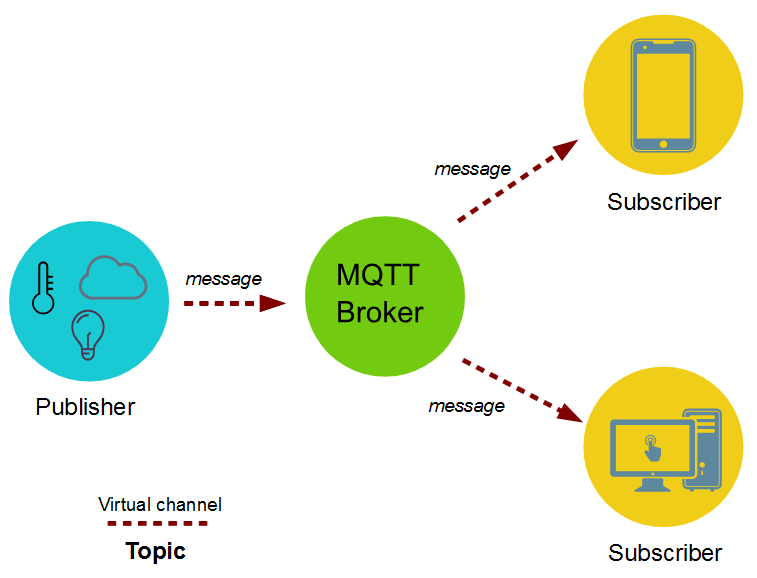
MQTT has unique features you can hardly find in other protocols, like:

* It’s a lightweight protocol. So, it’s easy to implement in software and fast in data transmission.
* It’s based on a messaging technique.
* Minimized data packets. Hence, low network usage.
* Low power usage. As a result, it saves the connected device’s battery.

**Working:**

 MQTT is based on clients and a server. The server is the one who is responsible for handling the client’s requests of receiving or sending data between each other. MQTT server is called a broker and the clients are simply the connected devices.

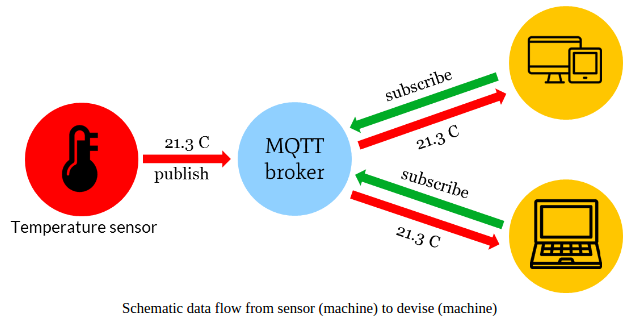
* When a device (a client) wants to send data to the broker, we call this operation a “publish”.
* When a device (a client) wants to receive data from the broker, we call this operation a “subscribe”.
* Publish, is the process a device does to send its message to the broker.
* Subscribe, where a device does to retrieve a message from the broker.

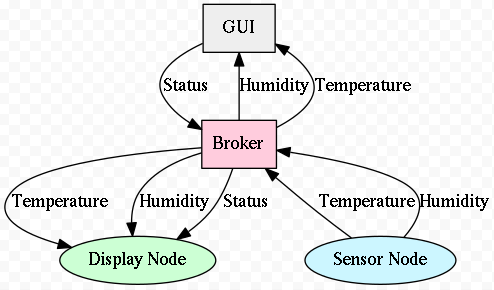


**Fig 6.9: MQTT Broker**

**6.3.1 Sending sensor data to client through MQTT protocol:**

The data such as temperature and humidity values from publisher is send to the **mqtt broker**. The broker role here is to take the message “temperature value” and deliver through a message to the subscriber phone or application.

**Fig 6.10:** **MQTT Schematic Data Flow**

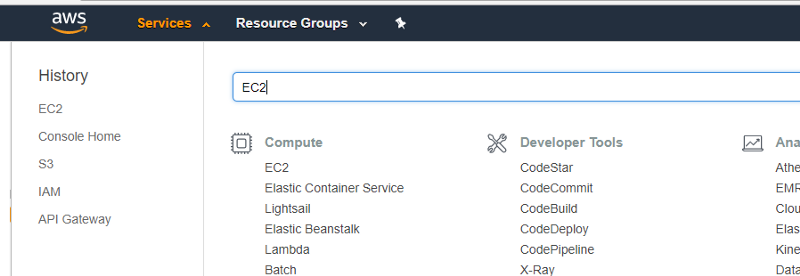


**Fig 6.11:** **MQTT Display INFO**

### 6.3.2 Configure MQTT

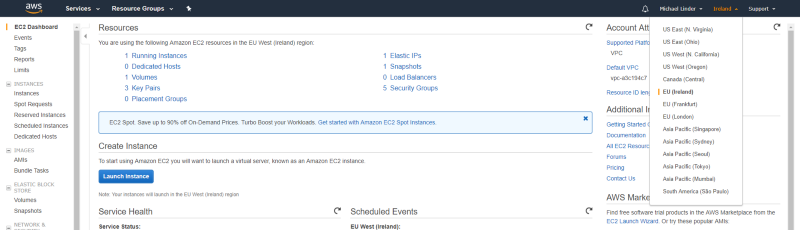
First of all, you need to have an AWS account. If you don’t have one, you can create a new account on AWS ([https://aws.amazon.com](https://aws.amazon.com/)) and get one-year computing, storage and several other services for free using the so called “Free Tier”.

1. Logon to the AWS Console and then select EC2 in services Section



Select EC2

2. Select the preferred AWS region and then launch a new instance



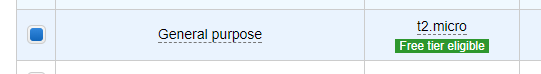
Select AWS region and launch

3. Select Ubuntu Server

https://cdn-images-1.medium.com/max/800/1*njCIH6d4m93BEb5xARdtNg.png

4. Select an instance type

In order to use the free tier contingent, it’s recommended to use t2.micro for testing purposes.



t2.micro selection

5. Configure security group

In this example following ports will be used:

**Step 1: Install Mosquitto**

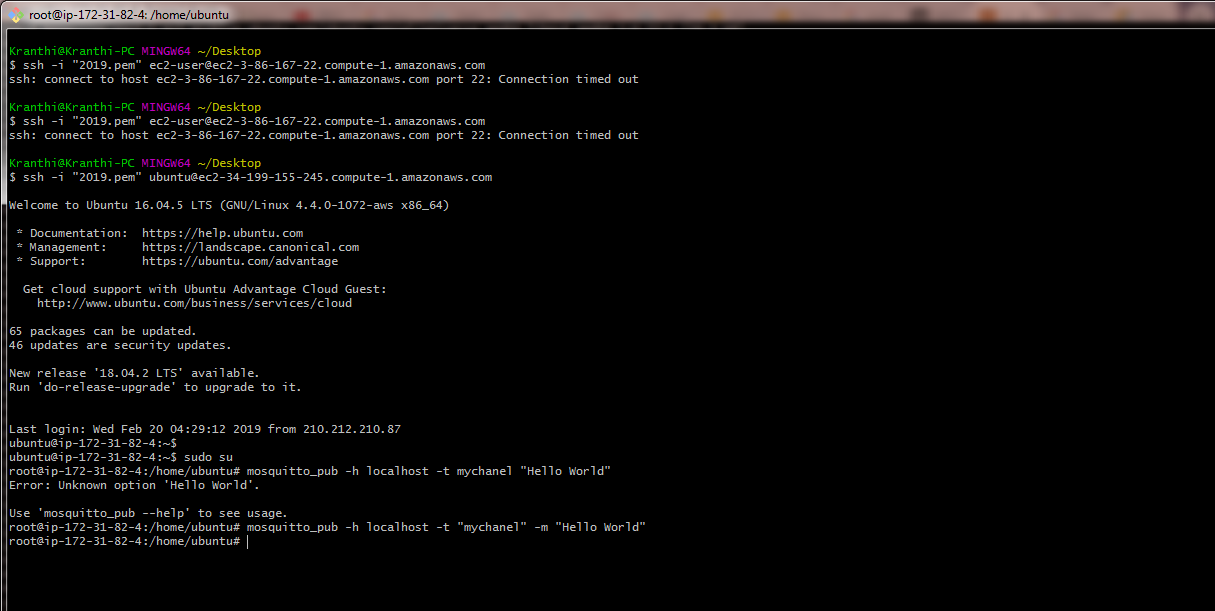
Log into the AWS Ubuntu 16/18 machine.

$ sudo apt-get update

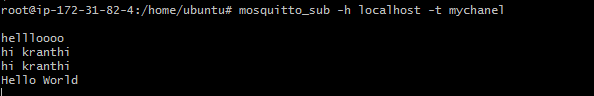
Install

$ sudo apt-get install mosquitto mosquitto-clients

The command above installs both the mosquitto broker and the publish / subscribe clients. The mosquitto broker is now installed and active. You can listen to declare any channel to subscribe and publish to test it



open the duplicate session for this Ubuntu, type the same command for publish



**6.4** **Database Creation**

This below code is the main source file Index.js to perform whole server operations. This index.js server calls the database table information and acts as an API to connect the web page and database.

We need to install npm, MySQL and express js to run this node js file.

**Npm (node packet manager):NPM** basically **is** the package manager for node. It helps with installing various packages and resolving their various dependencies. It greatly helps with your Node development.

**How it is helpful in our project**: Using this npm, we have specified all our project’s dependencies inside the package.json file, when any time we want to run our web server we can just **run npm install** and all the dependencies installed. Manually adding these files kills time and messy.

**MySQL**: AWS support MySQL to store the values into a database we used MySQL. We can add, delete, retrieve, and modify the values on a database.

**Express Js**: It is a web application framework for nodejs and provides simple API’s to build websites, web-apps, and backend. It is flexible as there are numerous modules available on **npm**, which can be directly plugged into Express.

The database created here is atmosphere dB, This atmospheredb contains atmosphere table with field names temperature, humidity and created time.

**6.4.1** **Setting Database path:**

**MySQL DB Creation**

#!/bin/bash

# this script is not compatible with dash shell; hence the above shebang is mandatory

# ------------------------------------------------------------------------- #

# This script shall create database and required users.

#INSTRUCTIONS:

#1.This script executes as mysql root

#2.change dbname, pwd values respectively

#3.Depending on VM & mysql installation edit the dbpath variable

#Scripts to be executed in the following order:

# 1.db\_scripts\_p1.sh script

# 2. create tables, foreign keys etc..

# 3.db\_scripts\_p2.sh script

# Created date : 26-Jan-18

# Initial Author : Sai

# ------------------------------------------------------------------------- #

# ---Variables to be modified at runtime. User permissions-------------------------------- #

**dbname=atmospheredb**

**owner\_user='root'**

**owner\_pwd='passw0rd'**

**root\_pwd='passw0rd'**

**dbpath="mysql -u root -e "**

#dbpath="$mysql --host=127.0.0.1 -u root -p$root\_pwd -e "

# ------------------------------------------------------------------------- #

# Here $dbname will call node js program, there given database name , it will be recalled and stored in $dbname# ------------------------------------------------------------------------- #

# creating databse steps

**$dbpath"CREATE DATABASE $dbname ;"**

# ------------------------------------------------------------------------- #

# creating OWNER user / grants steps

**$dbpath"CREATE USER '$owner\_user' IDENTIFIED BY '$owner\_pwd' ;"**

**$dbpath"GRANT ALL ON $dbname.\* TO '$owner\_user' WITH GRANT OPTION;"**

#$dbpath "ALTER USER '$owner\_user'@'localhost' IDENTIFIED WITH mysql\_native\_password BY '$owner\_pwd';"

**Table Values**

Resources to create db:

User.sql=>contains the sql table creation and data to be inserted into created table

create table atmospheretable(

id int auto\_increment,

temperature smallint,

humidity smallint,

created\_time datetime,

raining\_value smallint unsigned,

PRIMARY KEY(id)

);

create table productdetailtable(

id int auto\_increment,

fullname varchar(200),

color varchar(50),

mrp smallint,

discount smallint,

brand varchar(50),

waranty smallint,

podavailable boolean,

replaceperiod smallint,

category varchar(50),

PRIMARY KEY(id)

);

insert into productdetailtable(fullname,color,mrp,discount,brand,waranty,podavailable,replaceperiod,category) values('JBL wireless in-ear earphones ASDFGHJKL 13123123','Black',2500,20,'JBL',1,true,40,'Earphones');

insert into productdetailtable(fullname,color,mrp,discount,brand,waranty,podavailable,replaceperiod,category) values('Sony in-ear earphones ASDdfgd 1d3434f','Black',4300,10,'SONY',2,true,20,'Earphones');

insert into productdetailtable(fullname,color,mrp,discount,brand,waranty,podavailable,replaceperiod,category) values('JBL wireless in-ear earphones ASDFGHJKL 13123123','Black',2500,25,'boAt',0,false,40,'Headphones');

insert into productdetailtable(fullname,color,mrp,discount,brand,waranty,podavailable,replaceperiod,category) values('JBL wireless in-ear earphones ASDFGHJKL 13123123','Black',2500,20,'JBL',1,true,40,'Earphones');

insert into productdetailtable(fullname,color,mrp,discount,brand,waranty,podavailable,replaceperiod,category) values('JBL wireless in-ear earphones ASDFGHJKL 13123123','Black',2500,20,'JBL',1,true,40,'Earphones');

insert into atmospheretable(temperature,humidity,created\_time,raining\_value) values(38.6,26,'2018-09-12',300);

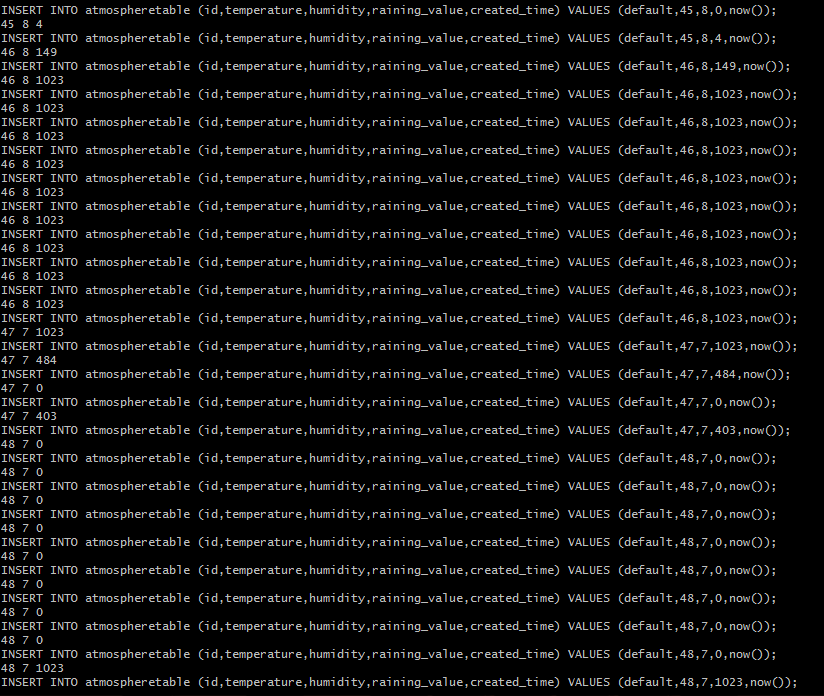
insert into atmospheretable(temperature,humidity,created\_time,raining\_value) values(46.9,12,'2018-09-12',255);

insert into atmospheretable(temperature,humidity,created\_time,raining\_value) values(37.6,77,'2018-09-12',280);

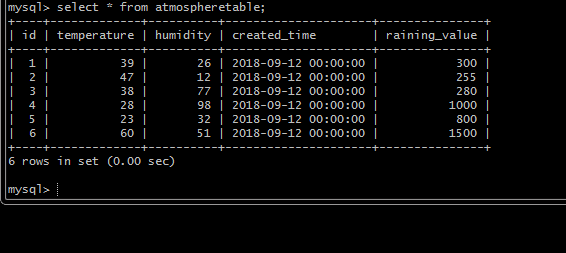
insert into atmospheretable(temperature,humidity,created\_time,raining\_value) values(28.4,98,'2018-09-12',1000);

insert into atmospheretable(temperature,humidity,created\_time,raining\_value) values(22.6,32,'2018-09-12',800);

insert into atmospheretable(temperature,humidity,created\_time,raining\_value) values(59.8,51,'2018-09-12',1500);



**Fig 6.9: Data Inserting to Database**



**Fig 6.10: Database Table**

**6.4.2** **Downloading and Installations:**

Initialsetupdeploy.sh (Script file for installing node js npm and creating wbsitedeploy directory)

#!/bin/bash

curl -sL https://rpm.nodesource.com/setup\_10.x | sudo -E bash -;

**yum install nodejs --enablerepo=nodesource -y;**

**npm install npm -y;**

**mkdir wbsite-deploy;**

**cd wbsite-deploy;**

**npm init -y;**

**npm install express --save;**

**npm install mysql --save;**

**6.4.3** **Connecting database to website:**

**Node index.js (Main java script file for )**

const express = require('express'); // express driver installation

const app = express();

const mysql = require('mysql'); //mysql driver installation

const dbcon = mysql.createConnection({

'host':'localhost', //creating a connection to a database

'port':3306,

'user':'root',

'database':'atmospheredb'

});

app.use(express.json());

app.use(express.static('public'));

dbcon.connect(function(err,res){

if(err) {

console.log("err",err);

}else{

console.log("success");

}

});

app.get('/api/getdata', function(req,res){//

getData(req, res);

});

app.post('/api/putdata', function(req,res){

putData(req.body, res);

});

function getData(req,res){

dbcon.query('select \* from atmospheretable order by created\_time desc limit 1;',function(error,result){

if(error)

{

res.send(error);

}else

{

res.send(result.length?result[0]:result);

}

});

}

//insert into atmosphere tables the values

function putData(req,res){

dbcon.query('insert into atmospheretable (temperature,humidity,created\_time) values (?,?,now()) ;',[req.temperature,req.humidity],function(error,result){

if(error) {

res.send(error);

}else {

res.send(result.length?result[0]:result);

}

});

}

app.listen(3001, () => console.log('Server running on port 3001')); //acknowledgement to the user that server is running.

**6.4.4 MQTT setup**

import ast

import paho.mqtt.client as mqtt

import ConfigParser

import simplejson as json

import logging

import datetime

import pymysql

LOG\_FILENAME = 'FeedbackServer.log'

logging.basicConfig(filename=LOG\_FILENAME,level=logging.DEBUG,format='%(asctime)s, %(levelname)s, %(message)s', datefmt='%Y-%m-%d %H:%M:%S')

class MQtt:

def \_\_init\_\_(self,host,port,subTopic,cur,db,timealive=60):

self.host = host

self.port = port

self.timealive = timealive

self.payload = None

self.subTopic = subTopic

self.pubTopic = "feedback-serv"

self.cur = cur

self.db = db

def \_\_on\_connect(self,client, userdata, flags, rc):

try:

print "Connected with result code "+str(rc)

if self.subTopic!=None:

(result,mid)= client.subscribe(self.subTopic)

print result

except Exception as e:

logging.error("The on\_connect error %s,%s"%(e,type(e)))

def \_\_on\_message(self,client, userdata, msg):

try:

data = msg.payload

message = ast.literal\_eval(data)

temperature = int(message["temperature"])

humidity = int(message["humidity"])

print temperature,humidity

insertStatement = "INSERT INTO atmospheretable (id,temperature,humidity,created\_time) VALUES (default,"+str(temperature)+","+str(humidity)+",now());"

print insertStatement

try:

self.cur.execute(insertStatement)

except Exception as e:

print e

except Exception as e:

print e

logging.error("The on\_message error %s,%s"%(e,type(e)))

def connect(self):

try:

self.mqttc = mqtt.Client()

self.mqttc.on\_connect = self.\_\_on\_connect

self.mqttc.on\_message = self.\_\_on\_message

self.mqttc.connect(self.host,self.port,self.timealive)

if self.subTopic != None:

print "Hu"

self.mqttc.loop\_start()

self.mqttc.loop\_forever()

except Exception as e:

logging.error("The connect error %s,%s"%(e,type(e)))

def send(self,message):

try:

(result,mid) = self.mqttc.publish(self.pubTopic,message,2)

return result

except Exception as e:

print e

logging.error("The send error %s,%s"%(e,type(e)))

if \_\_name\_\_ == '\_\_main\_\_':

# Mysql details

host = "localhost"

database = "atmospheredb"

table = "atmospheretable"

db = pymysql.connect(host="localhost",db="atmospheredb",autocommit=True)

cur = db.cursor()

# MQTT DETIALS

host = "18.232.46.180"

port = 1883

subTopic = "dht-resp"

mq = MQtt(host,port,subTopic,cur,db)

mq.connect()

**6.4.5 Storing Atmosphere Details into Database table**

import pymysql

import time

host = "localhost"

db = pymysql.connect(host="localhost",db="atmospheredb",autocommit=True)

cur = db.cursor()

temperature = 64

humidity = 36

print temperature,humidity

insertStatement = "INSERT INTO atmospheretable (id,temperature,humidity,created\_time) VALUES (default,"+str(temperature)+","+str(humidity)+",now());"

print insertStatement

try:

cur.execute(insertStatement)

time.sleep(10)

except Exception as e:

print e

sqlQuery = "select \* from atmospheretable;"

#Fetch all the rows - for the SQL Query

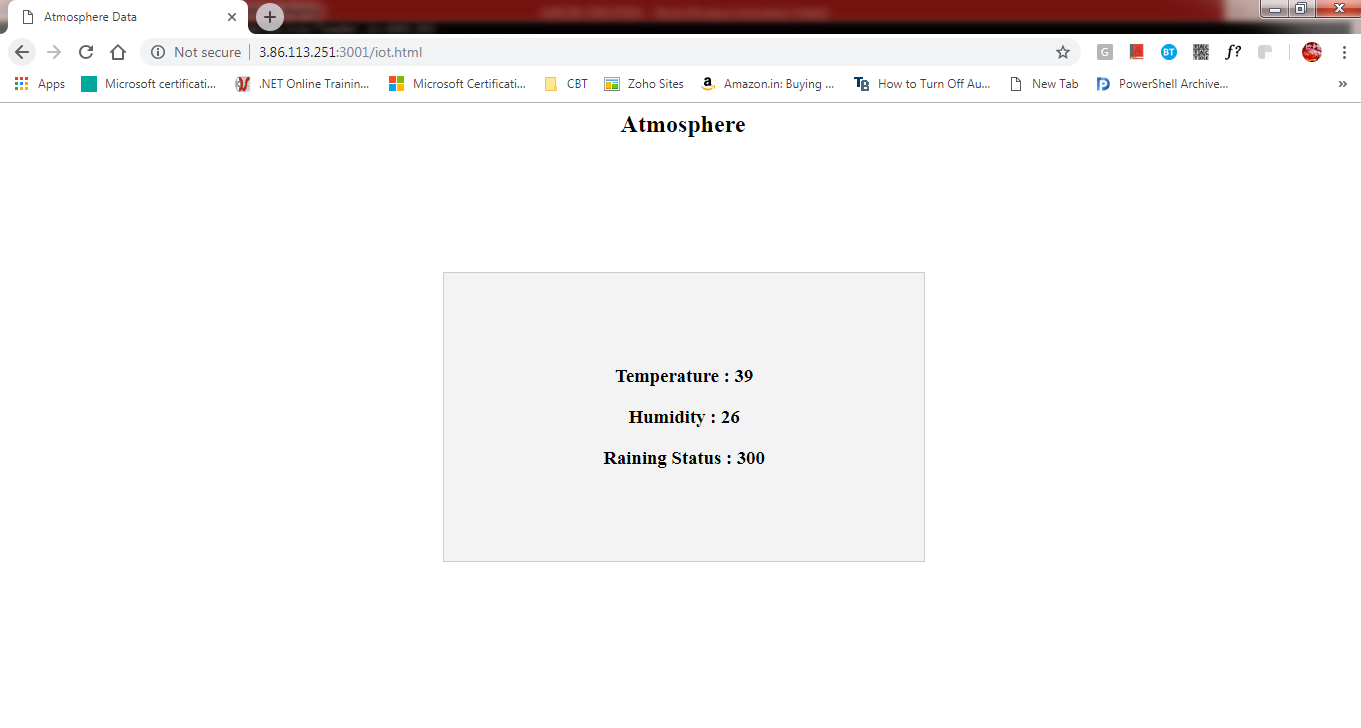
cur.execute(sqlQuery)

rows = cur.fetchall()

for row in rows:

print(row)

**IoT Web Page**



**CHAPTER-7**

**AMAZON EC2**

**7.1** **Introduction**

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

**7.2** **Features of Amazon EC2**

Amazon EC2 provides the following features:

1. Virtual computing environments, known as instances
2. Preconfigured templates for your instances, known as Amazon Machine Images (AMIs), that package the bits you need for your server (including the operating system and additional software)
3. Various configurations of CPU, memory, storage, and networking capacity for your instances, known as instance types
4. Secure login information for your instances using key pairs, Storage volumes for temporary data that's deleted when you stop or terminate your instance, known as instance store volumes
5. Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as Amazon EBS volumes
6. Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as regions and Availability Zones
7. A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using security groups
8. Metadata, known as tags, that you can create and assign to your Amazon EC2 resources
9. Virtual networks you can create that are logically isolated from the rest of the AWS cloud, and that you can optionally connect to your own network, known as virtual private clouds (VPCs)

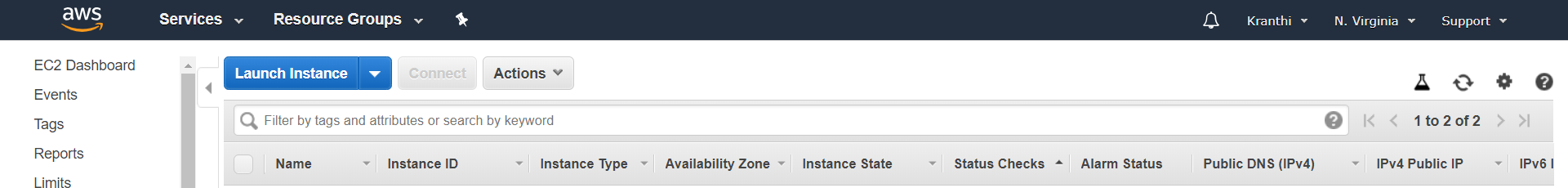
**7.3 How to launch an EC2 instance on AWS:**

Now we need to deploy this web portal on our EC2 instance to make this portal available for multiple users who wants to make use of.

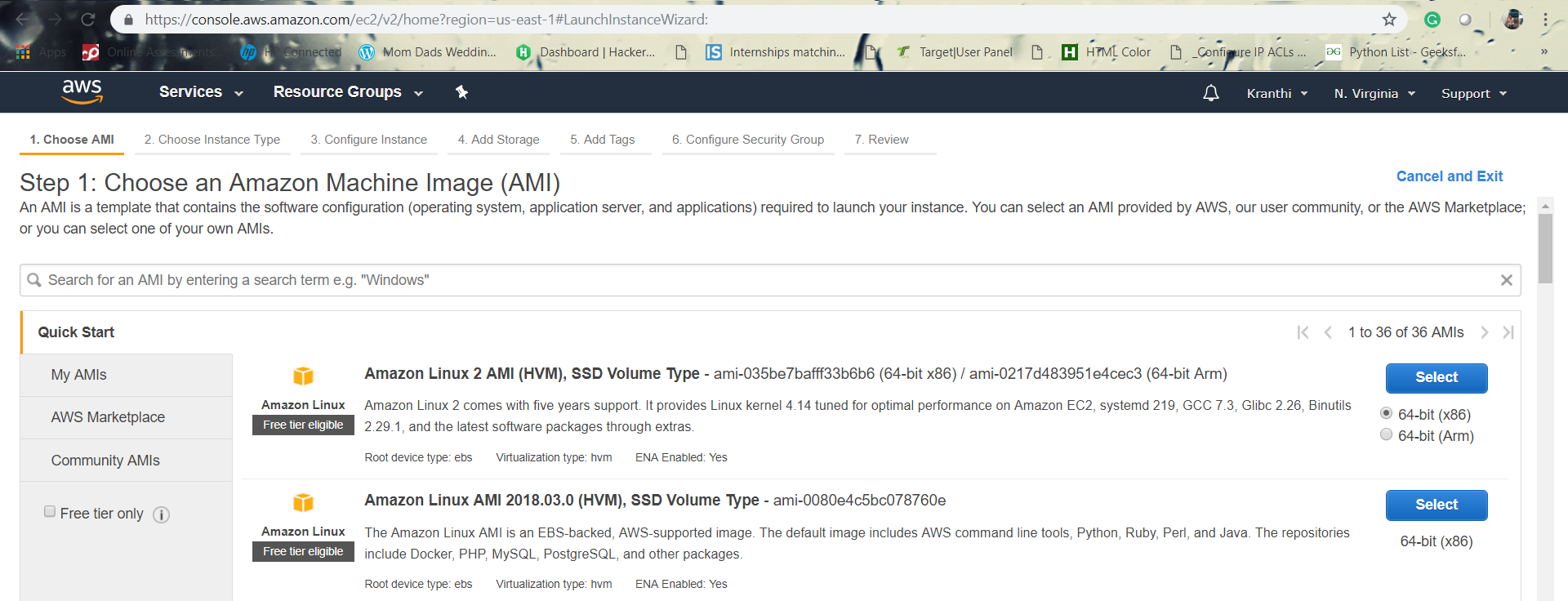
Creation of Instance in AWS Console:

Step 1: Choose the Instance option in AWS EC2.

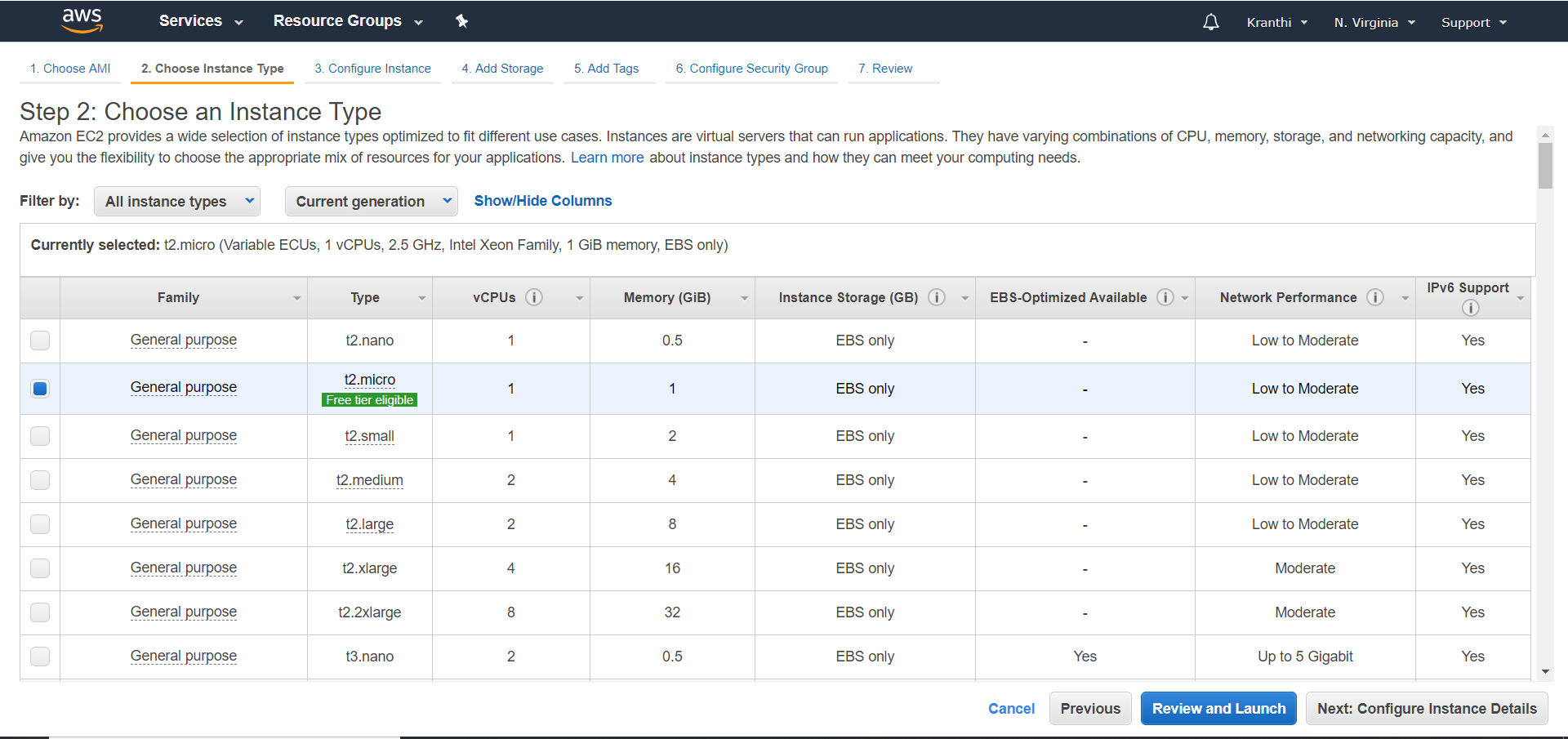
Step 2: Launch Instance that was on the top right side of the screen displayed.



Step 3: Choose an Amazon Machine Image(AMI).We choose (**Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type** - ami-0080e4c5bc078760e) free tier to launch our web portal.



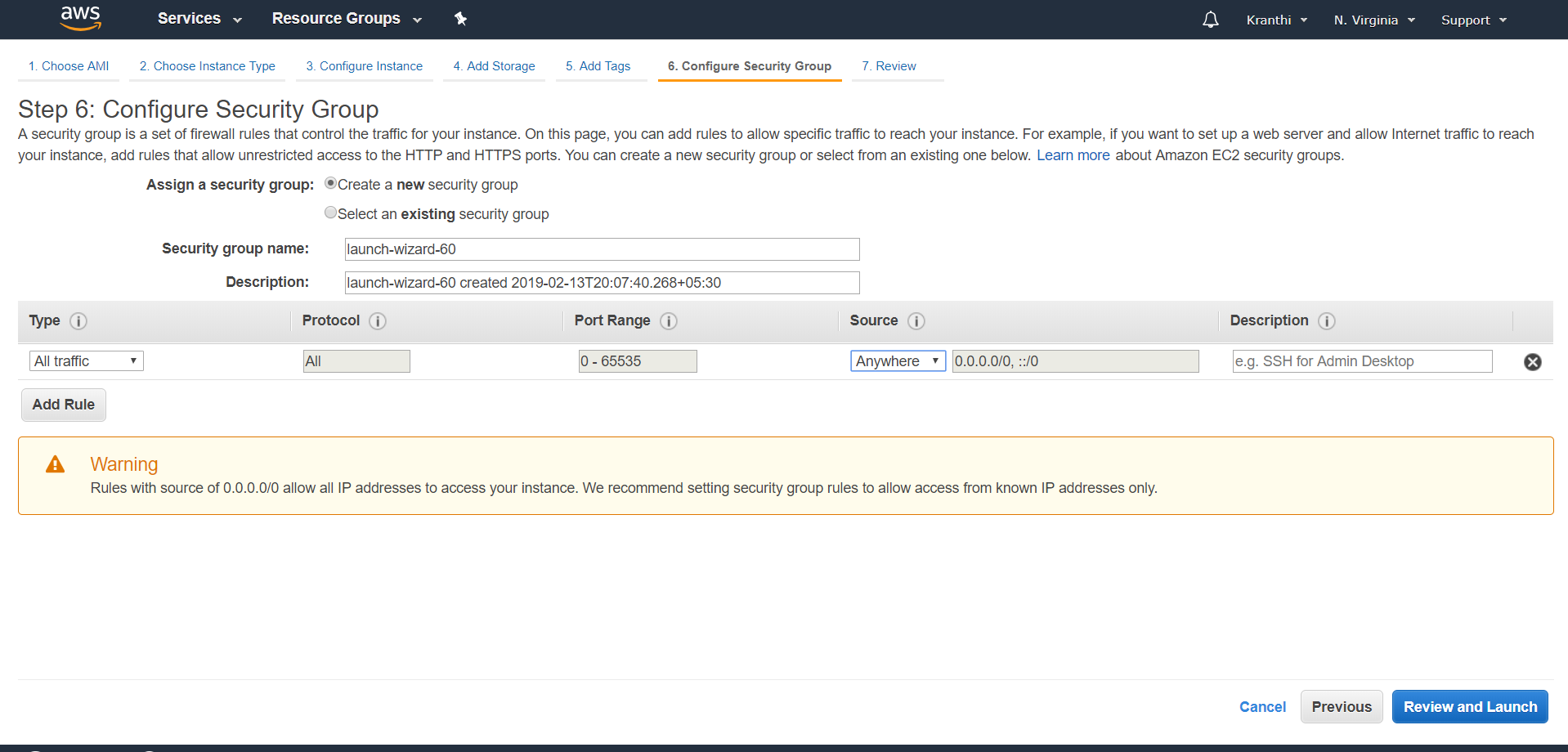
Step 4: Now choose an Instance type which fits to different use cases. Instances are Virtual Web servers that run our application. We choose (**:** t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)) for general purpose.



Step 5: 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.

Step 6: Add security groups

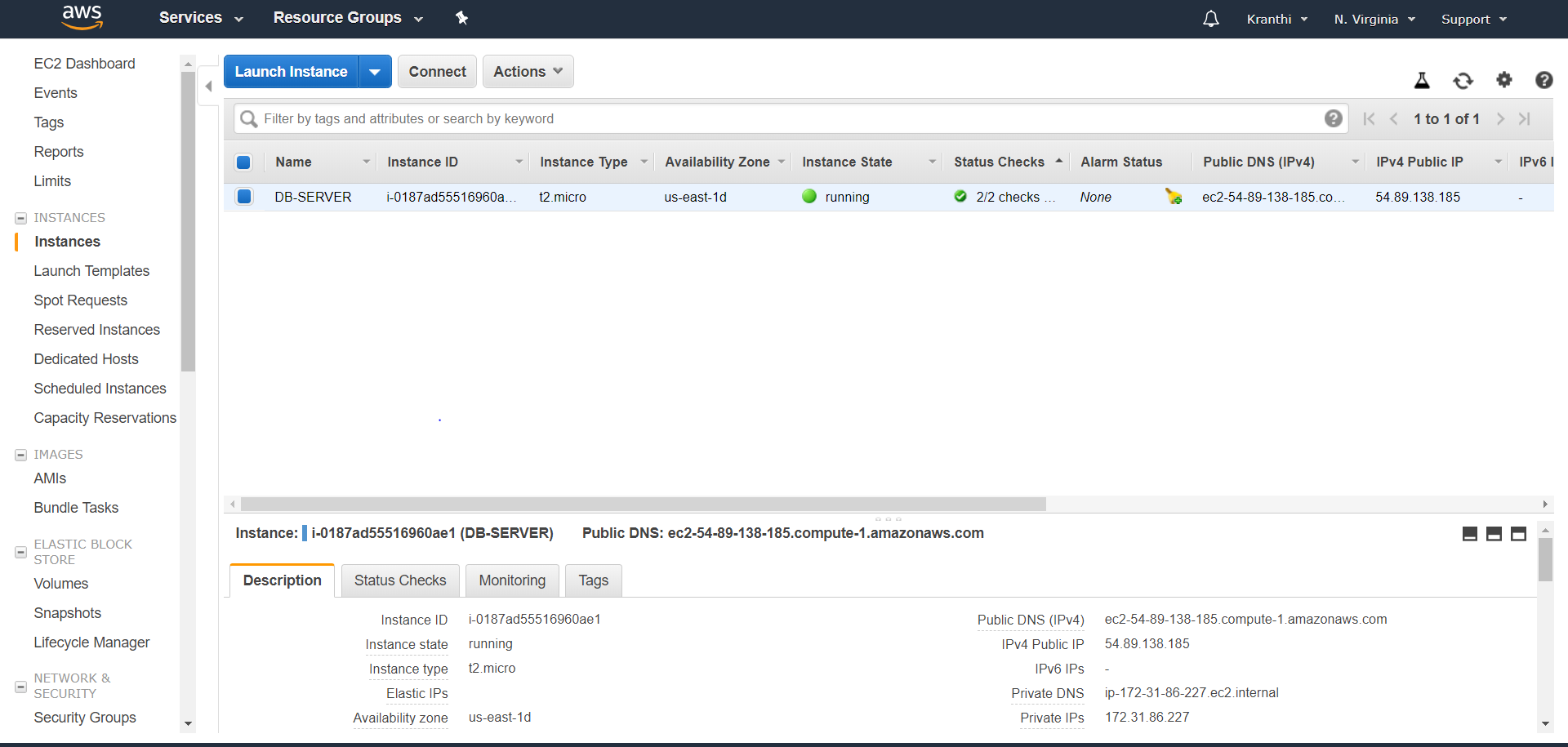


Step 7: On the **Review Instance Launch** page, choose **Launch**.

When prompted for a key pair, select **Choose an existing key pair**, then select the key pair that you created when getting set up.

Alternatively, you can create a new key pair. Select **Create a new key pair**, enter a name for the key pair, and then choose **Download Key Pair**.

This is the only chance to save the private key file, so be sure to download it. Save the private key file in a safe place. You'll need to provide the name of your key pair when launching an instance and the corresponding private key each time you connect to the instance.



1. Now launch the web portal on the created EC2 Instance. To deploy the web portal, we need to have a little idea about LINUX.
2. Install and run Gitbash, to manage ec2 instances.
3. Install filezilla. To move our web code files from desktop to cloud EC2 instance.

**7.3.1 Execute the following commands on Gitbash console:**

1. To connect EC2 instance remotely use the below command.

$ssh -i "2019.pem" [ec2-user@ec2-3-83-103-44.compute-1.amazonaws.com](mailto:ec2-user@ec2-3-83-103-44.compute-1.amazonaws.com)



**Fig 7.1: Git Bash Console**

**7.3.2 Procedure for running scripts on server:**

Sudo su //user permissions to access the EC2

Ls

Cd Resources/

ls

chmod 700 db\_setup.sh

chmod 700 initialSetupwensiteDeploy.sh

yum install mysql-server

service mysqld start

./db\_setup.sh

mysql -u root

mysql> show databases;

[root@ip-172-31-84-75 Resources]# **ls**

db\_setup.sh initialSetupWensiteDeploy.sh User.sql

[root@ip-172-31-84-75 Resources]# **mysql atmospheredb <User.sql**

[root@ip-172-31-84-75 Resources]# **mv initialSetupWensiteDeploy.sh /home/ec2-user/**

[root@ip-172-31-84-75 Resources]# **ls**

db\_setup.sh User.sql

[root@ip-172-31-84-75 Resources]# **cd ..**

[root@ip-172-31-84-75 ec2-user]# **ls**

as.pem initialSetupWensiteDeploy.sh Resources

[root@ip-172-31-84-75 ec2-user]# **./initialSetupWensieDeploy.sh**

## Installing the NodeSource Node.js 10.x repo...

[root@ip-172-31-84-75 ec2-user]# **cd db**

[root@ip-172-31-84-75 db]# **ls**

index.js public

[root@ip-172-31-84-75 db]# **mv index.js /home/ec2-user/website-deploy/**

[root@ip-172-31-84-75 db]# **mv public /home/ec2-user/website-deploy/**

[root@ip-172-31-84-75 db]# **ls**

[root@ip-172-31-84-75 db]# **cd ..**

[root@ip-172-31-84-75 db]# **npm install express –save**

[root@ip-172-31-84-75 db]# **npm install mysql --save**

[root@ip-172-31-84-75 ec2-user]# **cd wbsite-deploy/**

[root@ip-172-31-84-75 wbsite-deploy]# **ls**

index.js node\_modules package.json package-lock.json public

[root@ip-172-31-84-75 wbsite-deploy]# **node index.js**

**CHAPTER 8**

**ELASTIC LOAD BALANCER**

**8.1 Introduction**

Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, IP addresses, and Lambda functions. It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones. Elastic Load Balancing offers three types of load balancers that all feature the high availability, automatic scaling, and robust security necessary to make your applications fault tolerant.

These are classified into 3 types, they are Application Load Balancer, Network Load Balancer, and Classic Load Balancer. In our project we have used Application Load Balancer, this operates at the request level (layer 7), routing traffic to targets – EC2 instances, containers, etc. This provides advanced request routing targeted at delivery of modern application architectures. This simplifies and improves the security of our application, by ensuring that the latest SSl/TLS ciphers and protocols are used at all times.

* 1. **Advantages**

**High availability**

Elastic Load Balancing automatically distributes traffic across multiple targets – Amazon EC2 instances, containers and IP addresses – in a single Availability Zone or multiple Availability Zones.

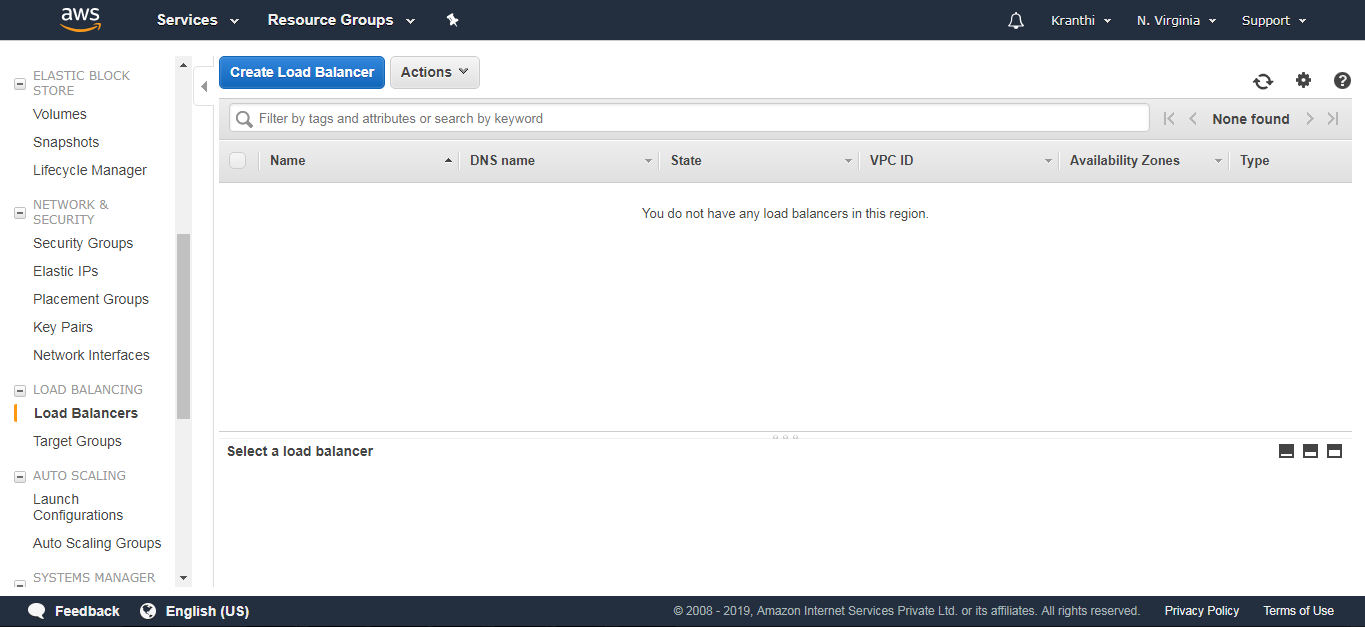
**Health Checks:**

An Application Load Balancer routes traffic only to healthy targets. With an Application Load Balancer, you get improved insight into the health of your applications in two ways: (1) health check improvements that allow you to configure detailed error codes from 200-499. The health checks allow you to monitor the health of each of your services behind the load balancer; and (2) new metrics that give insight into traffic for each of the services running on an EC2 instance.

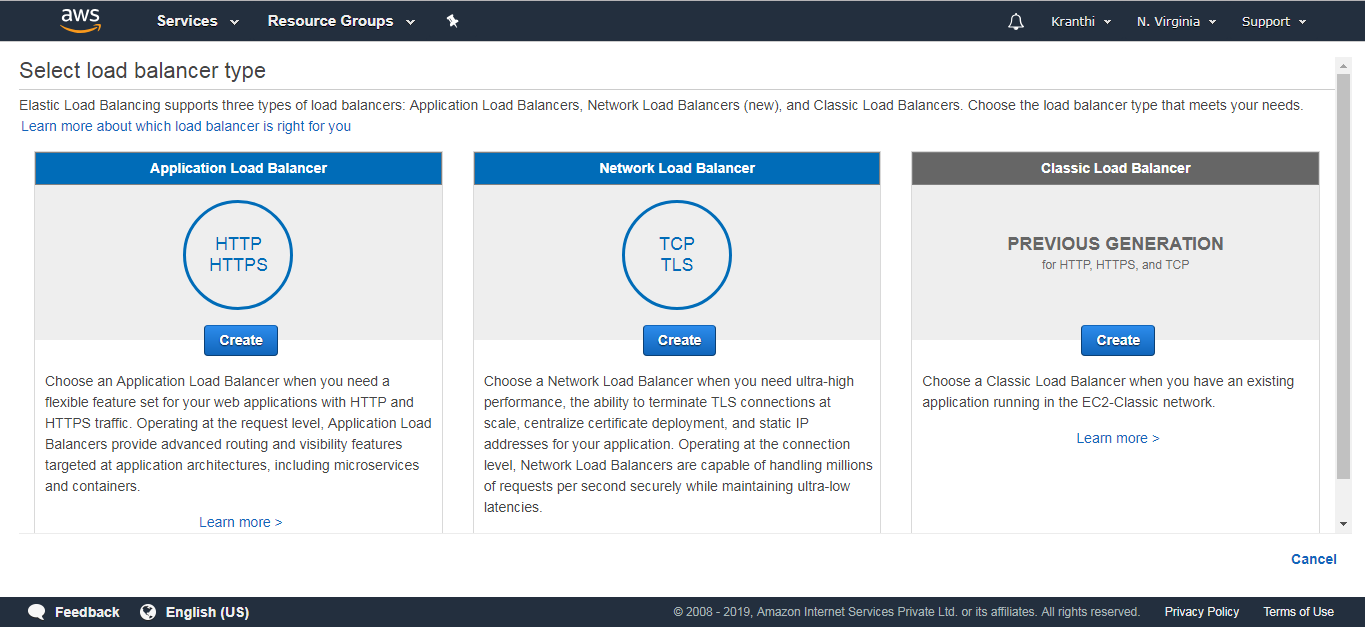
**8.3 How to add Load Balancer in AWS:**

Step 1: Login to the AWS Console.

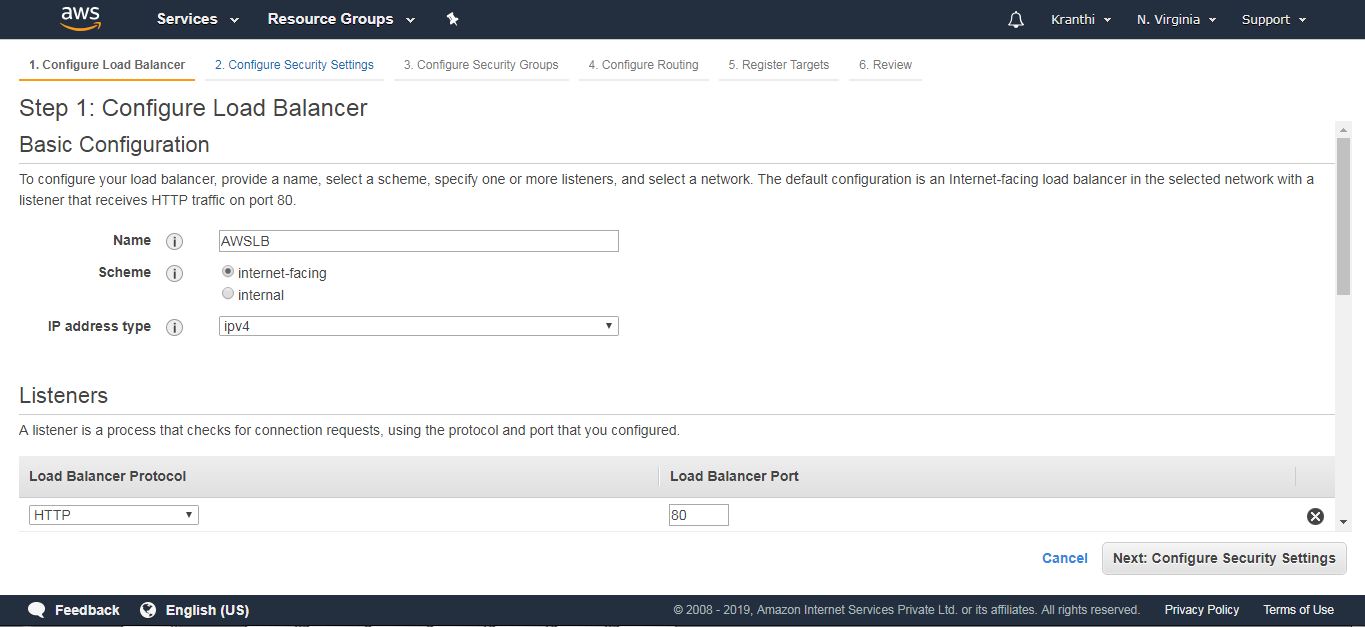
Step 2: Select EC2 Instances, there you find Load Balancer.

Step 3: Choose Create Load Balancer that appeared on the screen below.

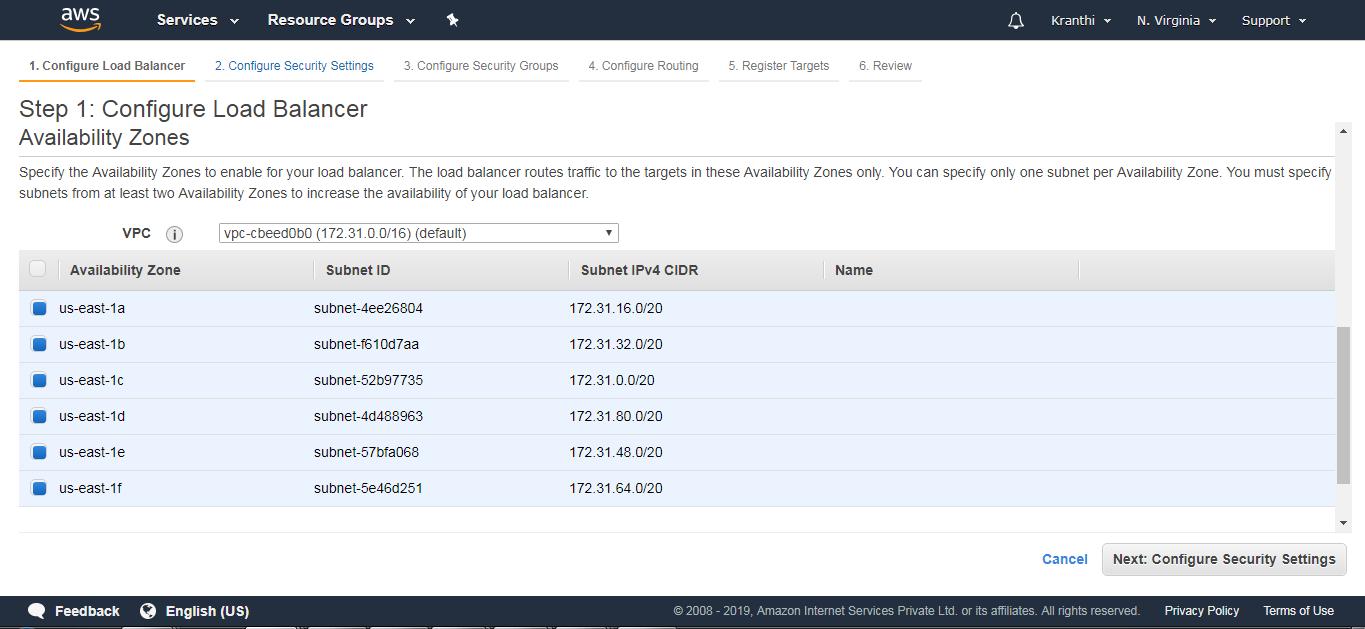
Step 4: Select Application Load Balancer and click on create.

 **Fig 8.1: Load Balancers**

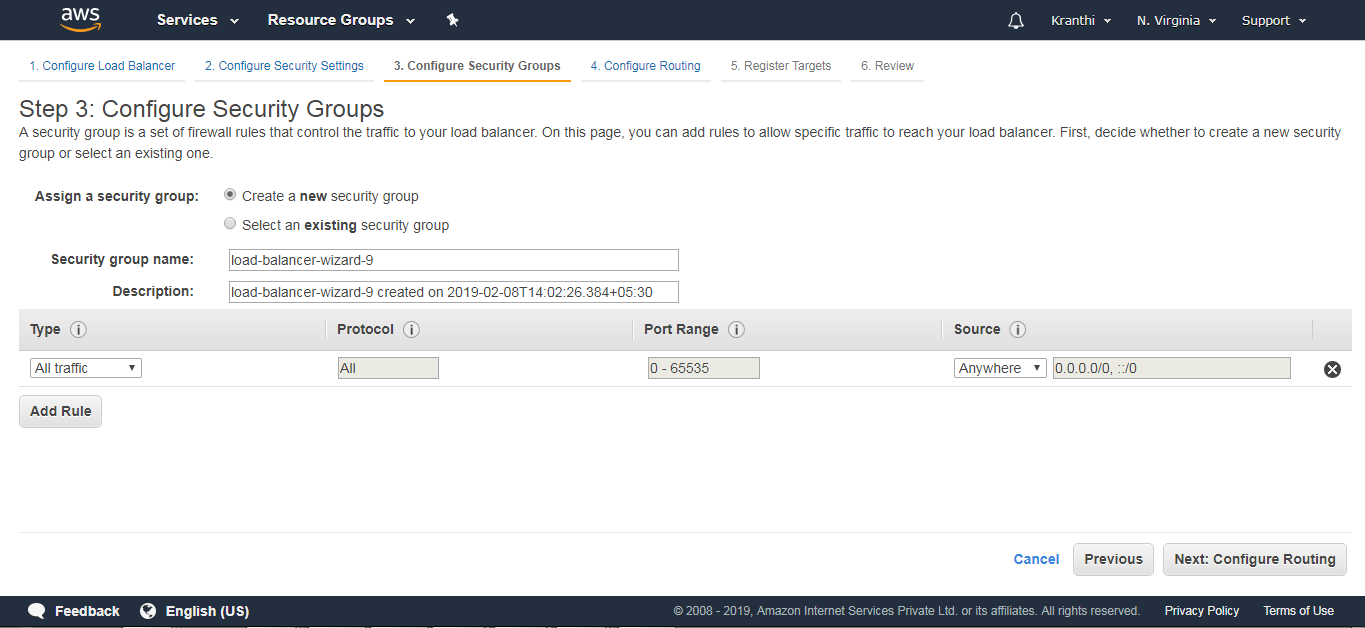
Step 5: Configure Load Balancer.



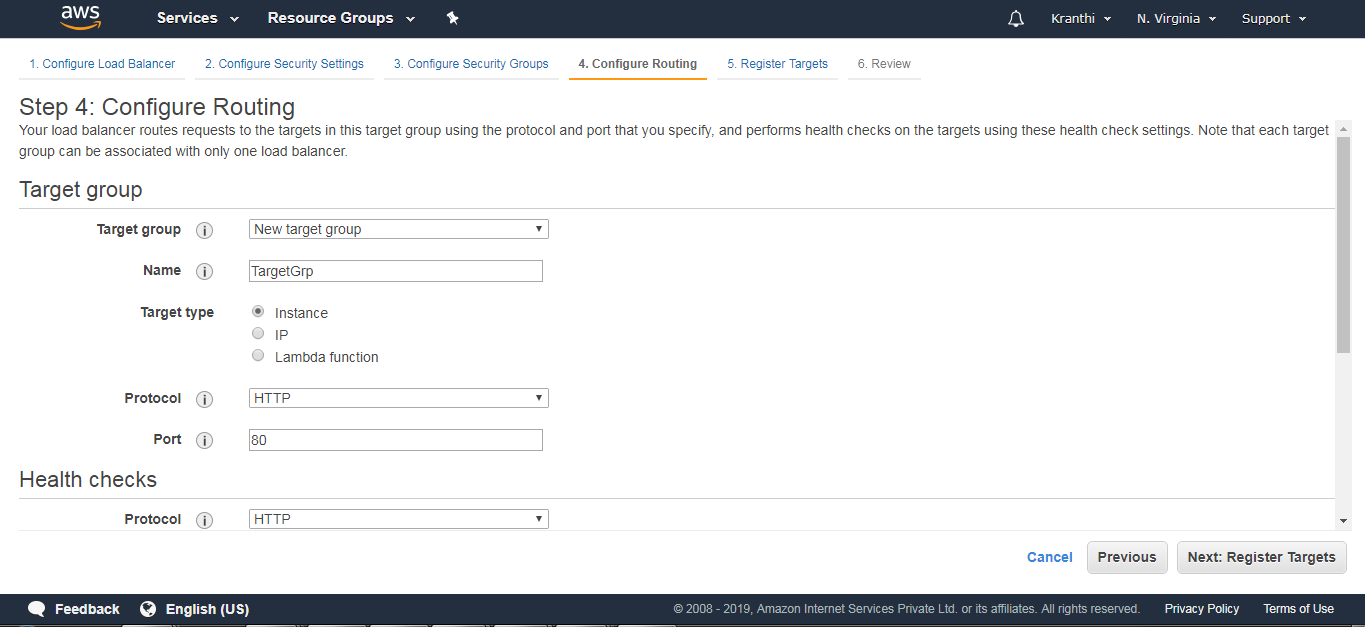
Add the availability Zones

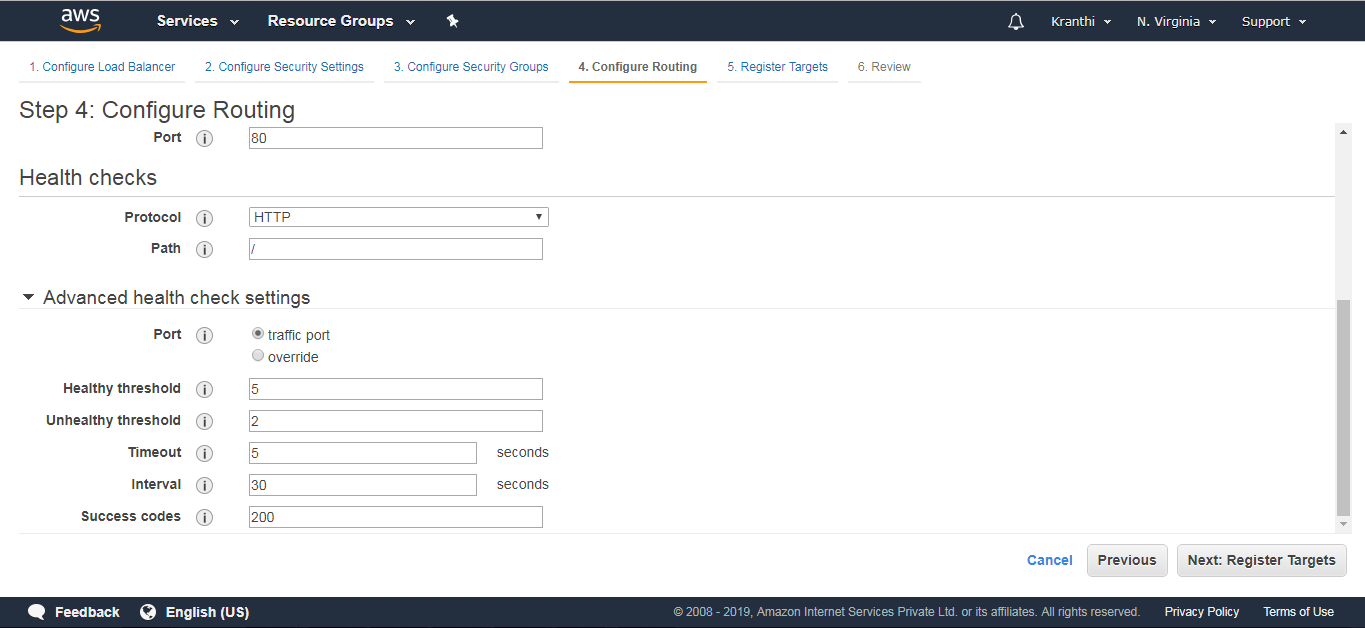


Step 6: Configure Security groups.

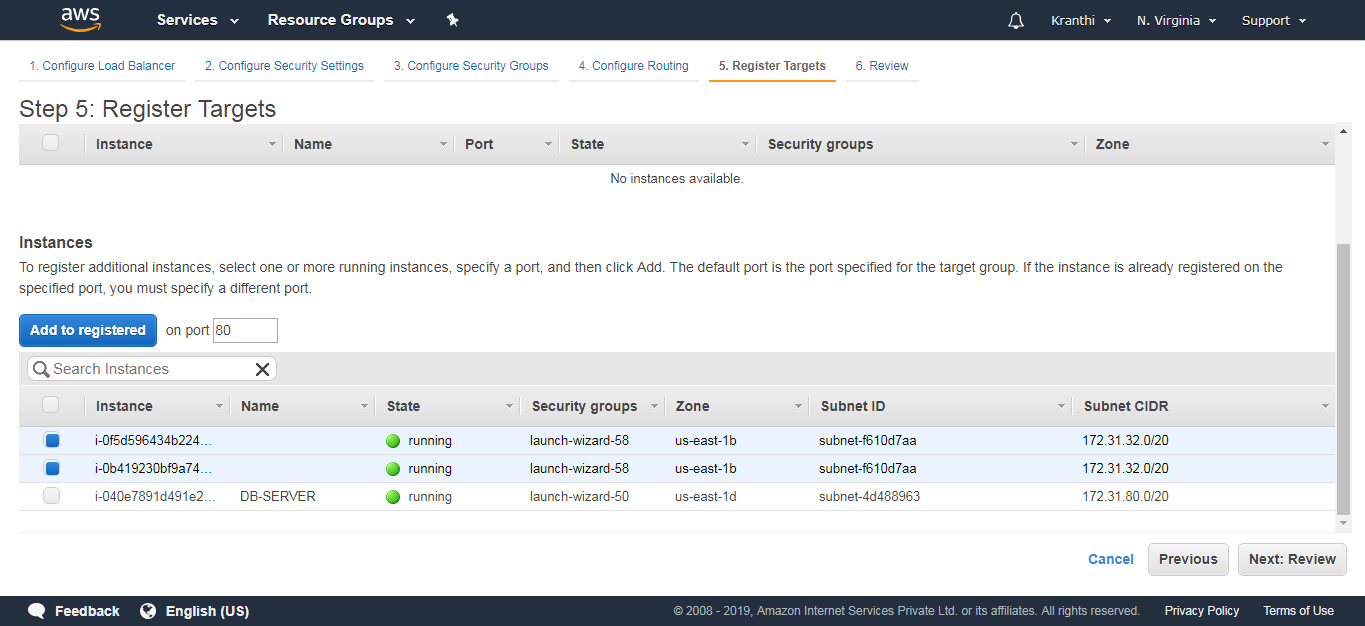


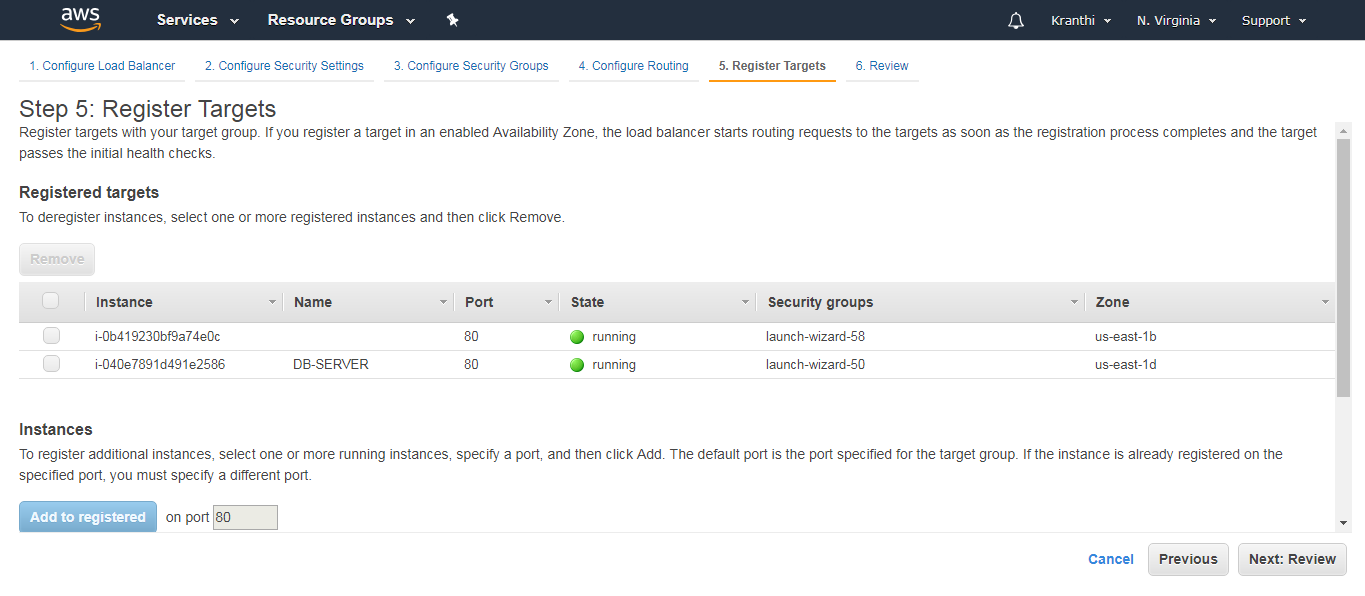
Step 7:Configure Routing



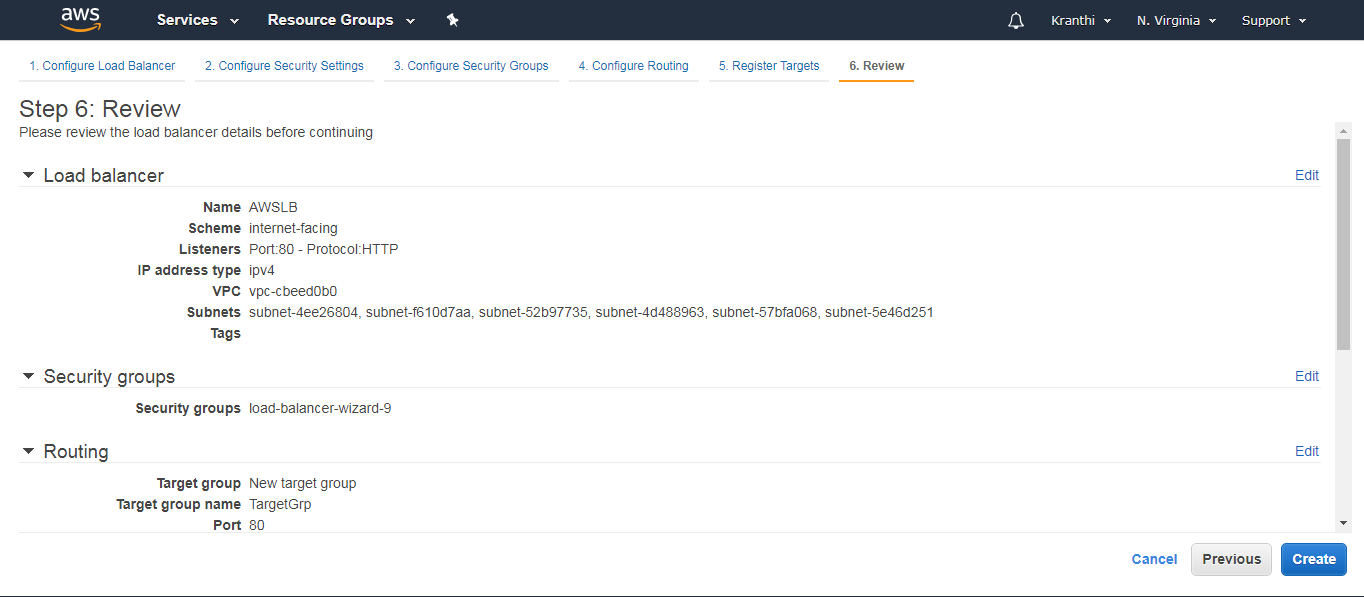


Step 8: Register Targets (nothing but EC2 instances that are created).

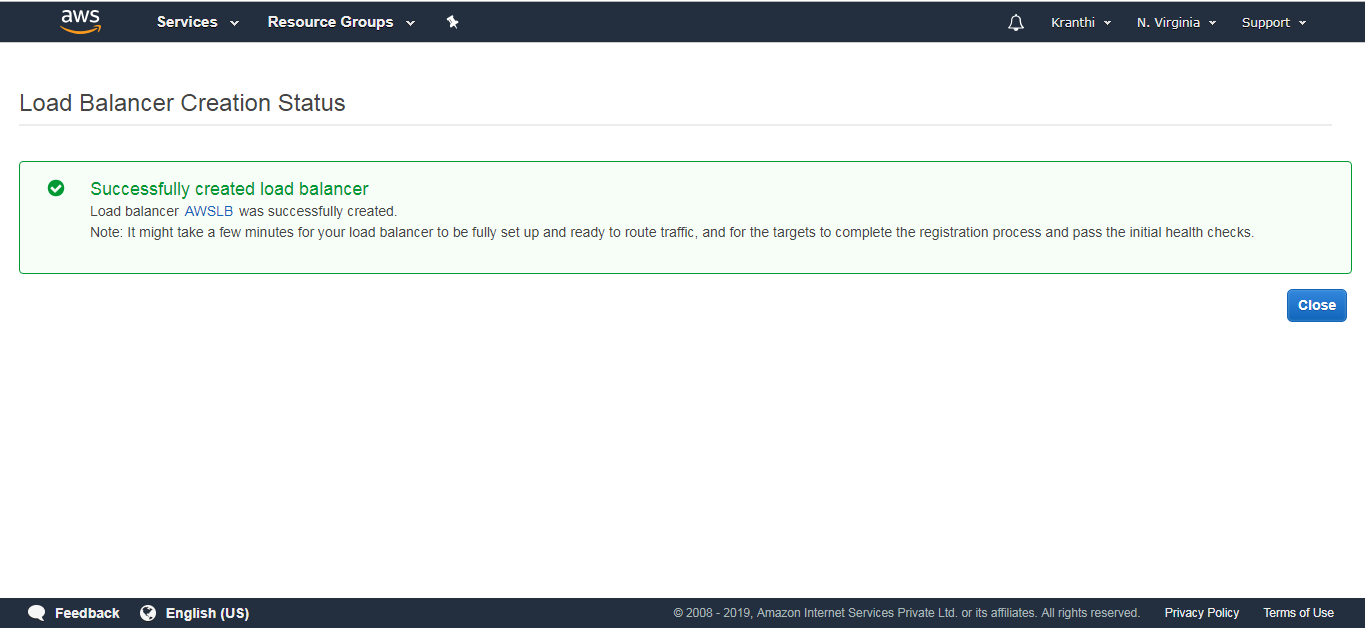




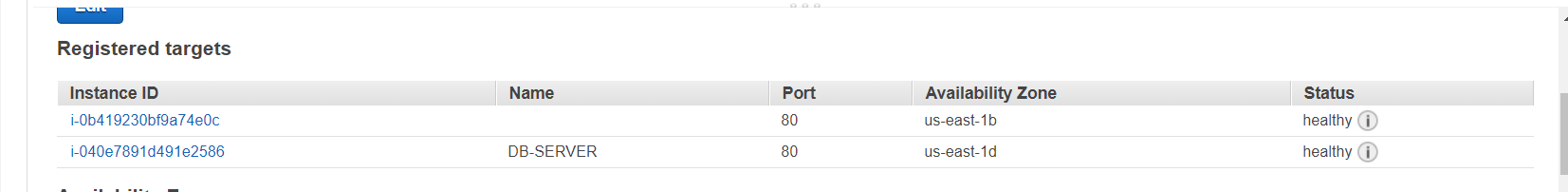
Review: View all the information of an Elastic Load Balancer.



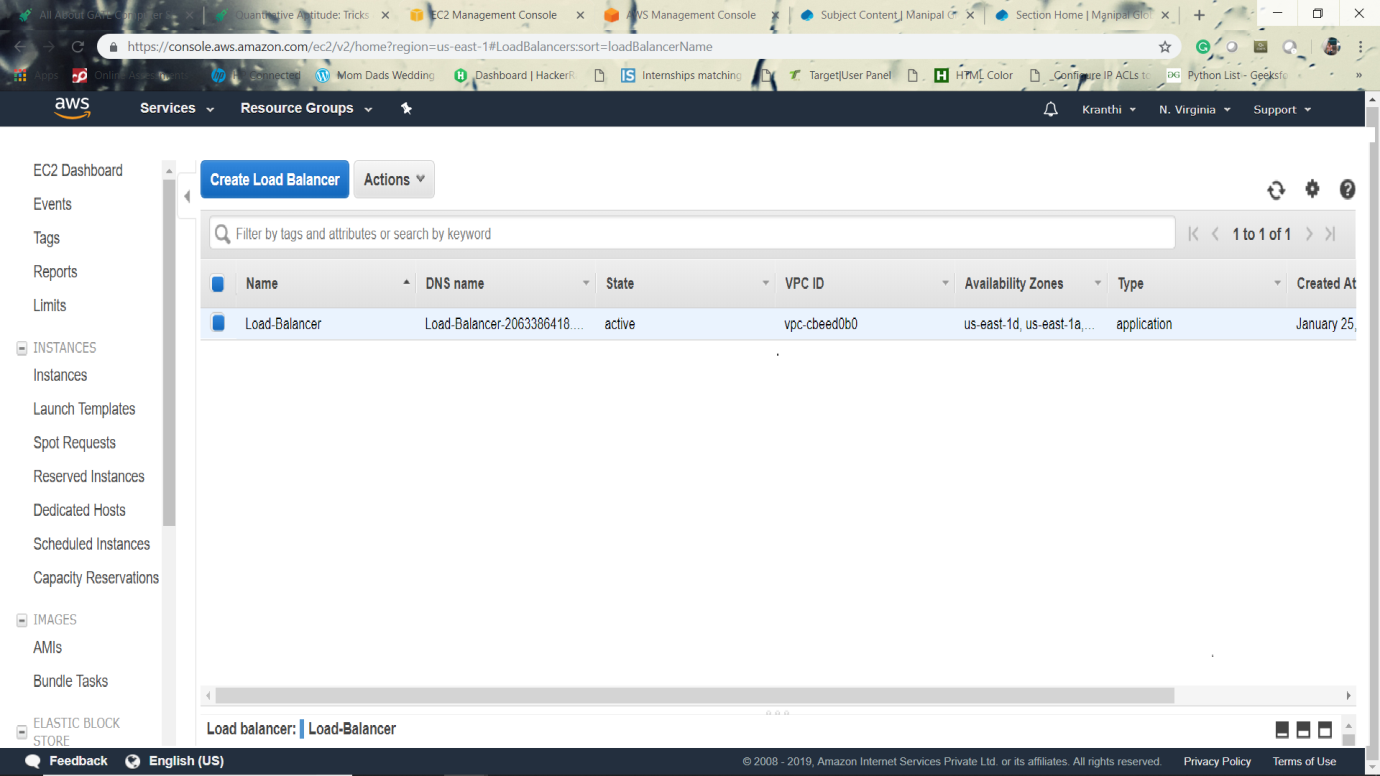
Step 9: We can now notice the notification that Load Balancer is created.



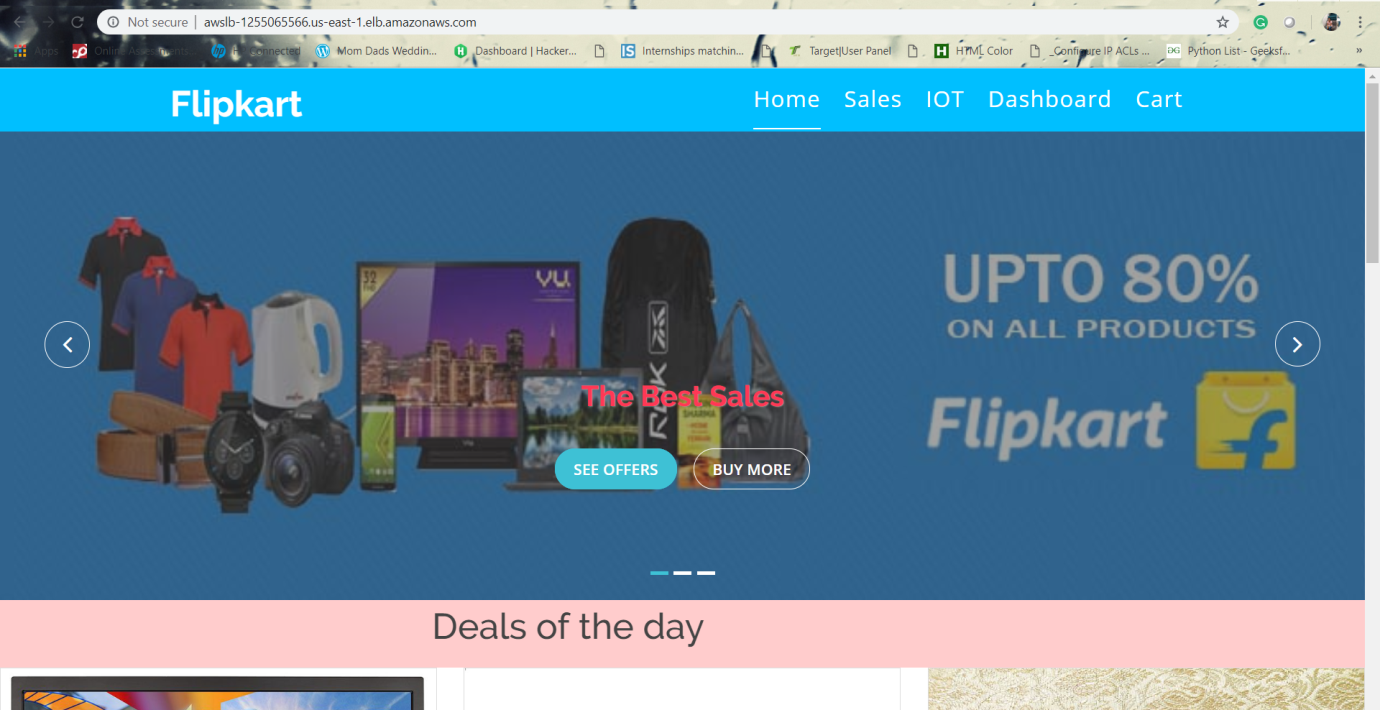
Check whether the web instances are healthy when added to a load balancer.



Step 10: This is the Load Balancer that we have created.



The web portal runs on a specified URL of Elastic Load Balancer



**Fig 8.2: Load Balancer URL of website**

**CHPATER 9**

**AUTO SCALING**

**9.1 Introduction**

An Auto Scaling group contains a collection of Amazon EC2 instances that share similar characteristics and are treated as a logical grouping for the purposes of instance scaling and management. If you might want to increase the number of instances in that group to improve the performance of the application. Or, you can decrease the number of instances to reduce costs when demand is low. Use the Auto Scaling group to scale the number of instances automatically based on criteria that you specify. You could also maintain a fixed number of instances even if an instance becomes unhealthy. This automatic scaling and maintaining the number of instances in an Auto Scaling group is the core functionality of the Amazon EC2 Auto Scaling service.

There are two tasks to be followed for Auto-Scaling, they are

1. Create a launch template

2. Create Auto-Scaling group.

**9.2 Create a Launch Template**

**To create a launch template for an Auto Scaling group**

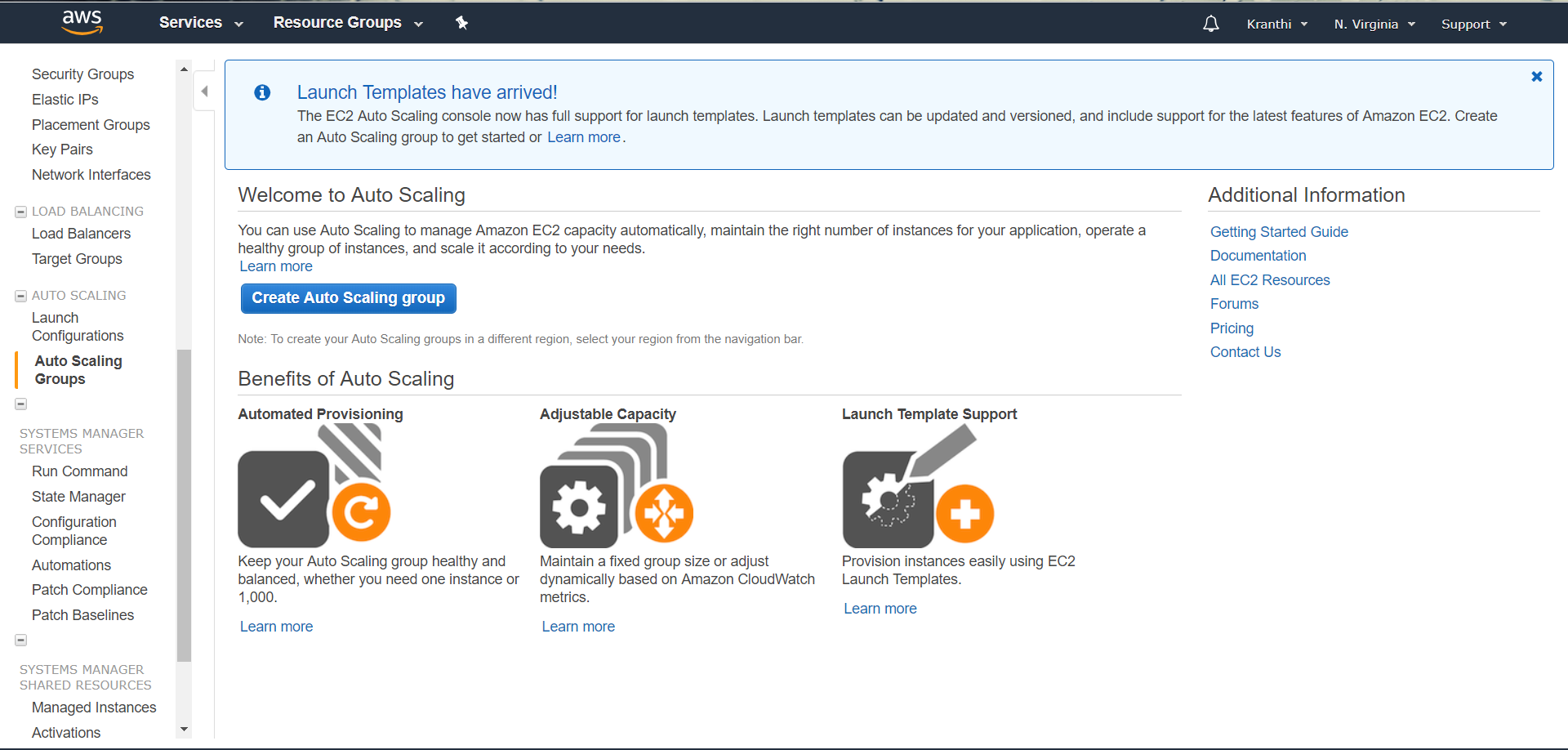
Step 1: Open the Amazon EC2 console.

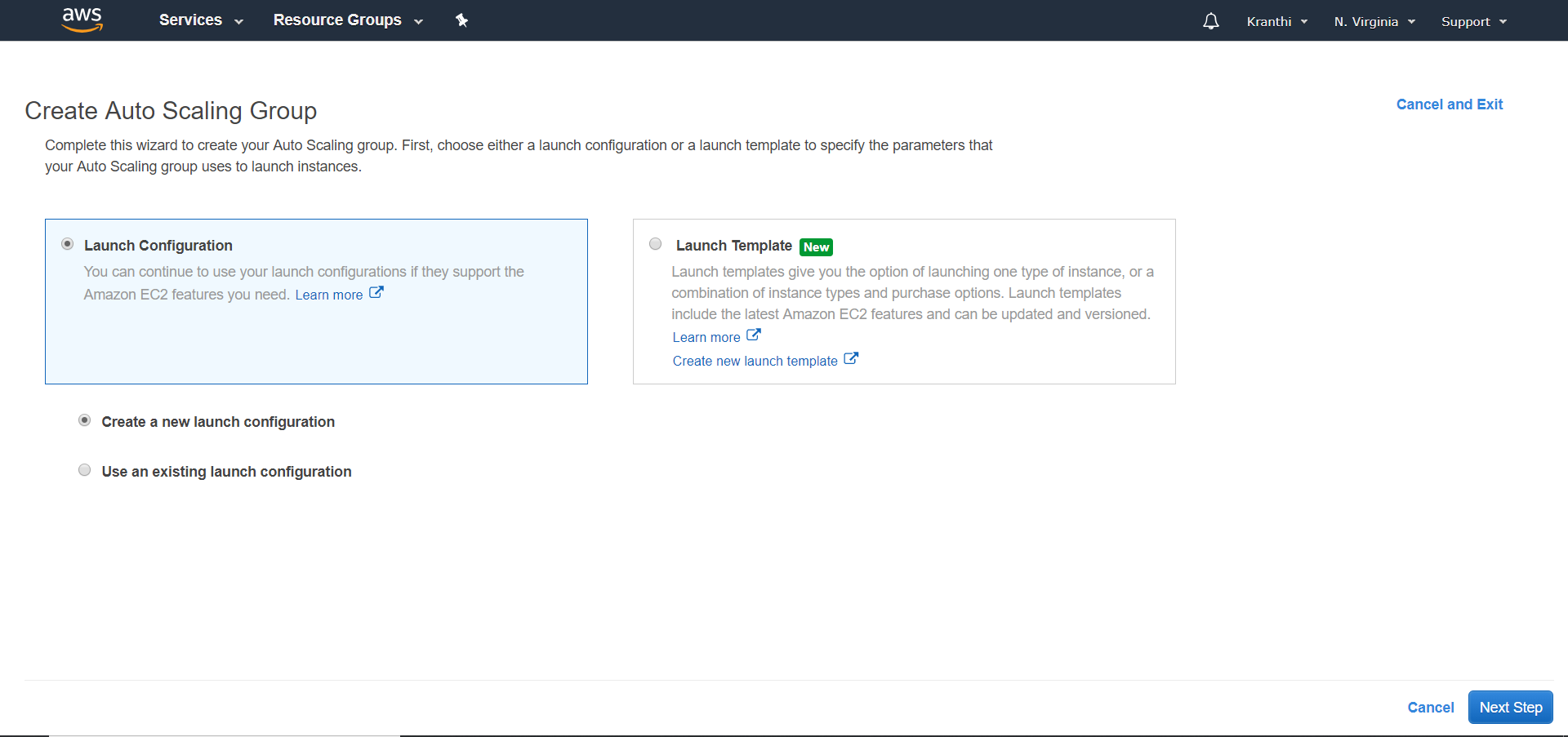
Step 2: On the navigation bar, select a region. The Amazon EC2 Auto Scaling resources that you create are tied to the region you specify.

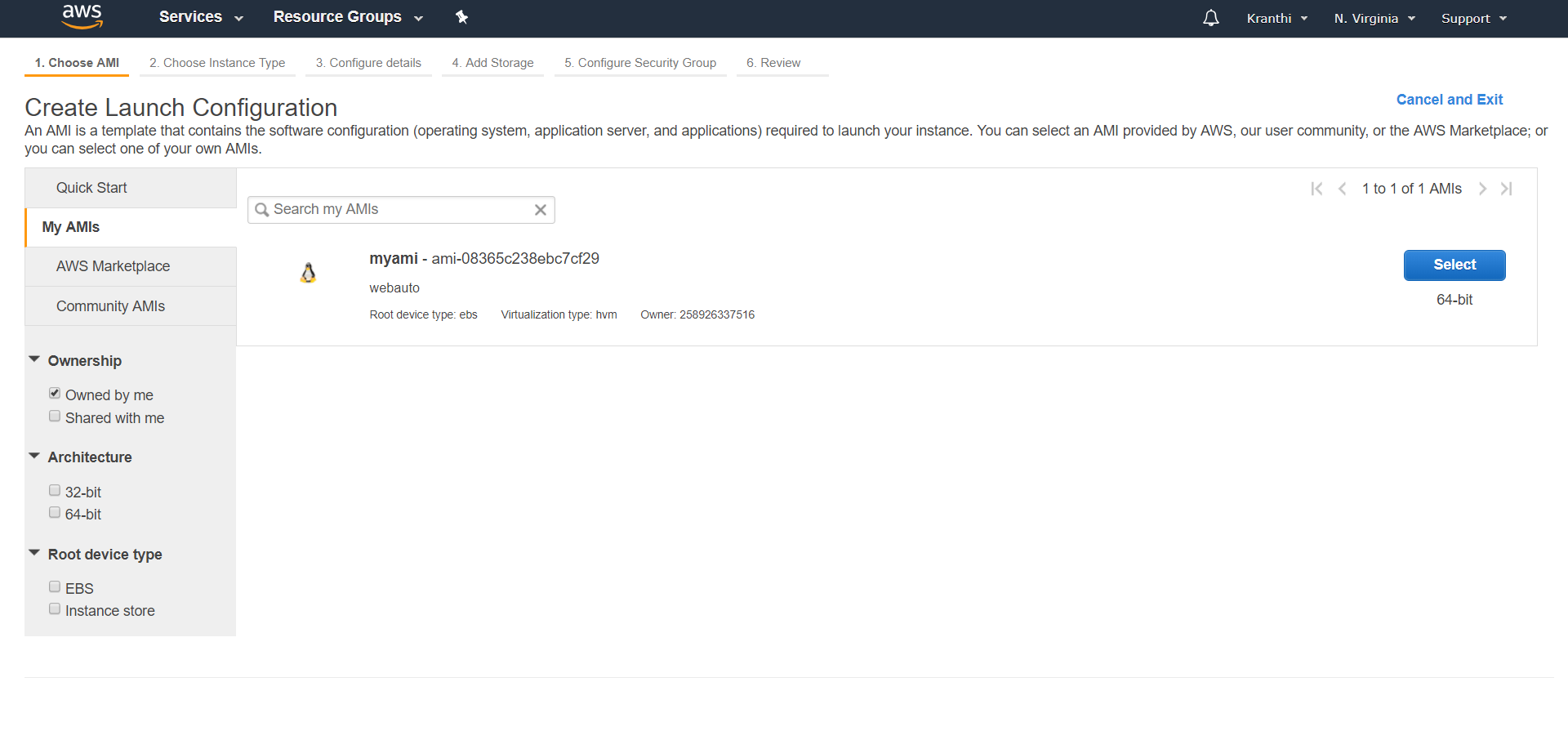
Step 3: On the navigation pane, choose Instances, Launch Templates.

Step 4: Choose Create launch template.

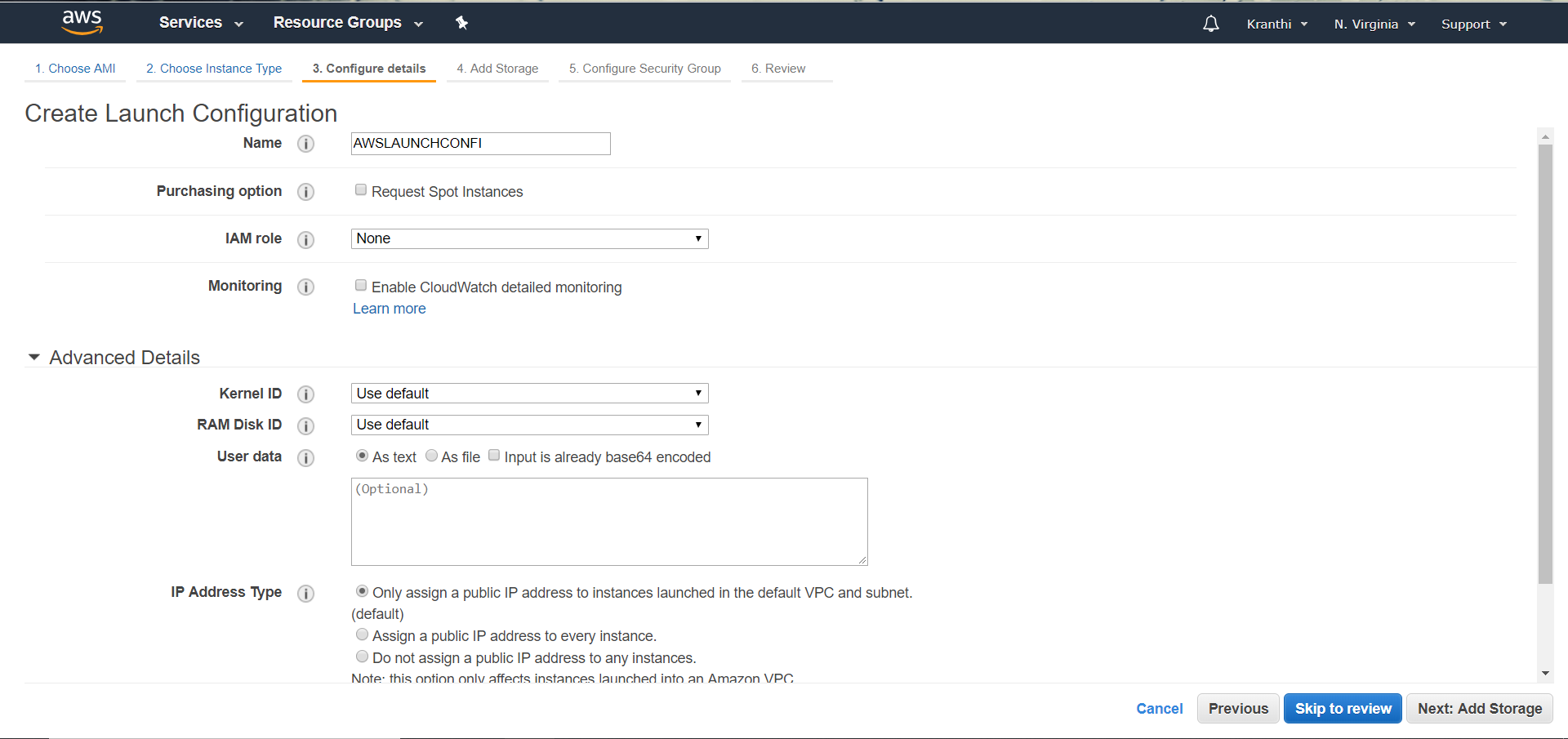
Step 5: Create Auto Scaling group.



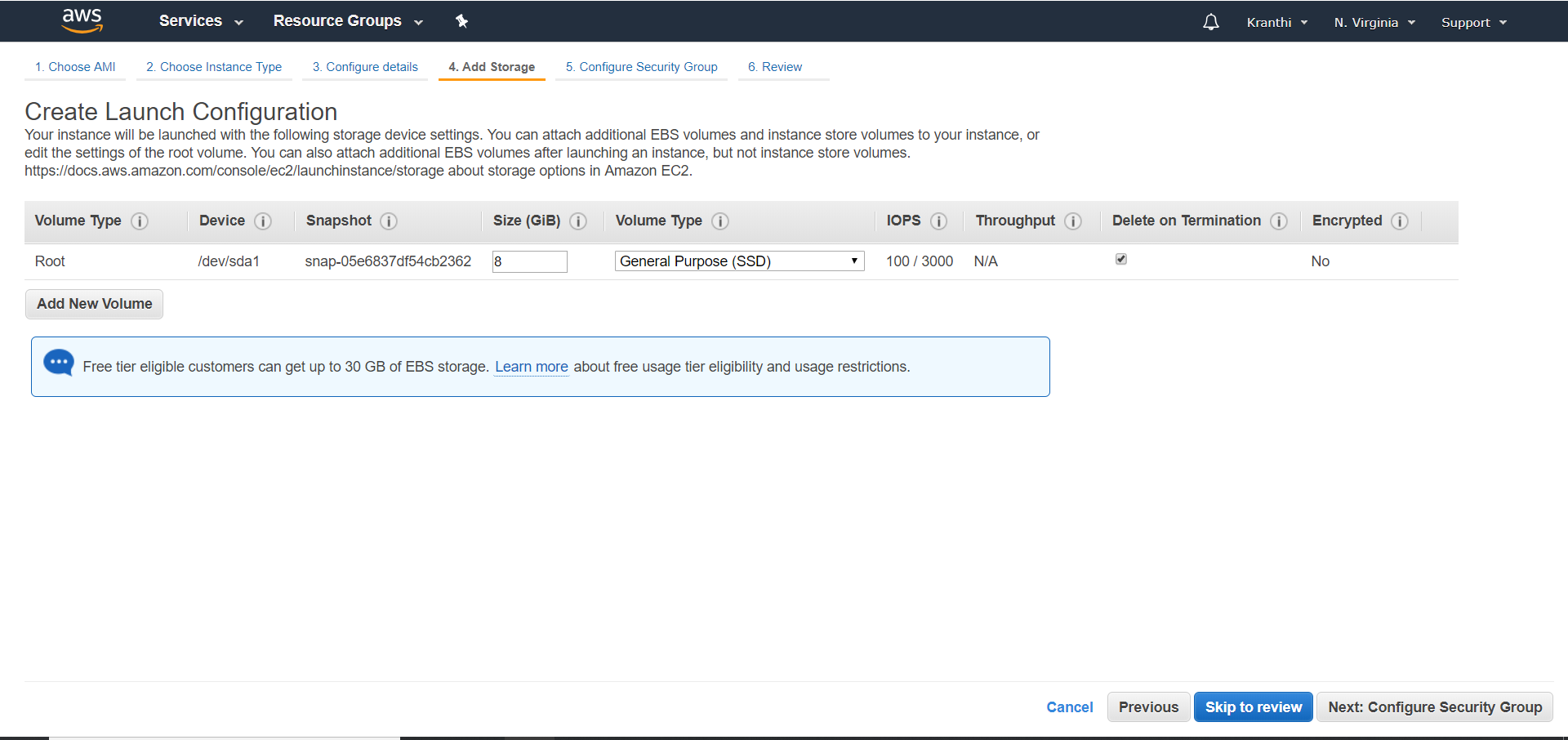
****An AMI is a template that contains the software configuration to launch the instances.

****

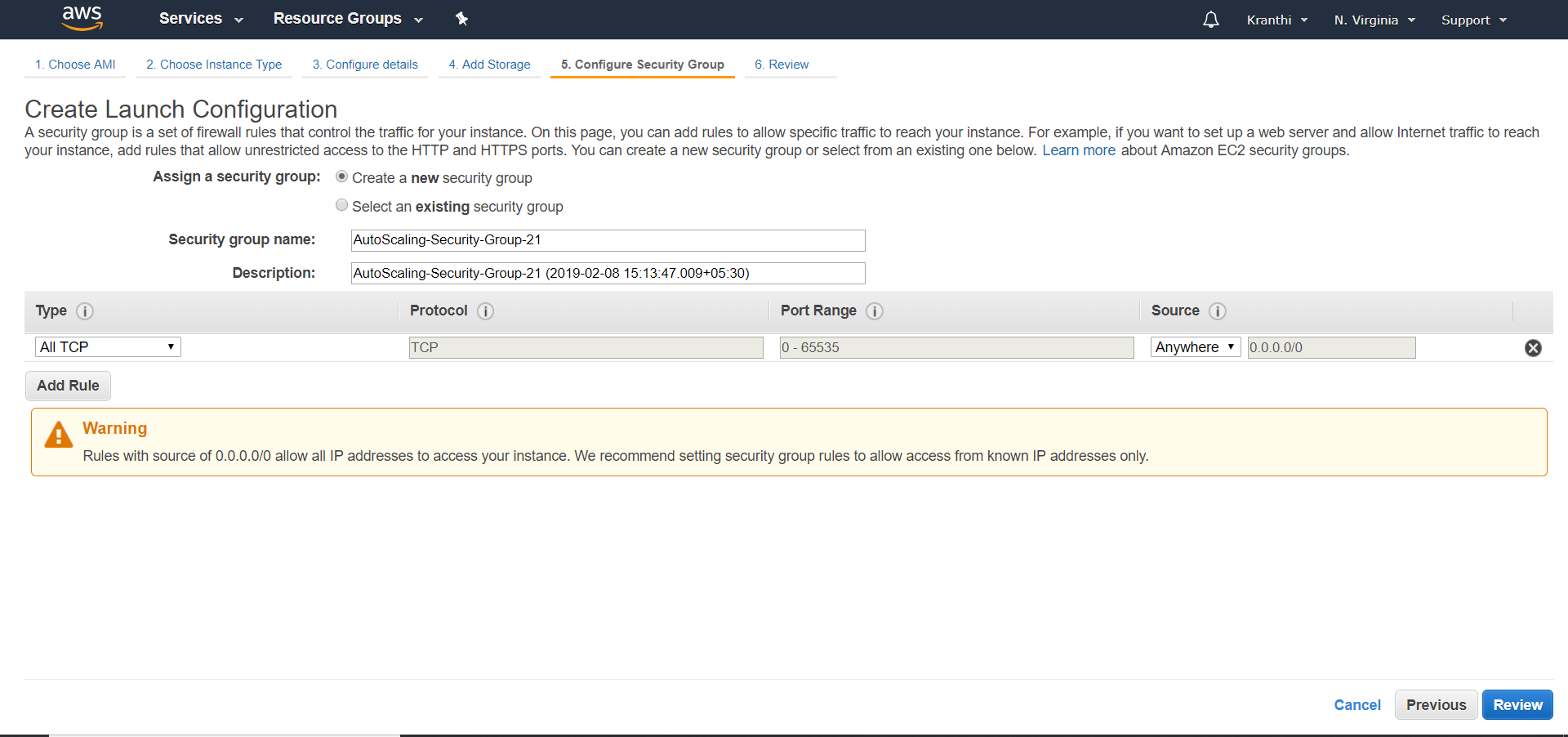
Configure the Launch configuration, set the name to display.

****

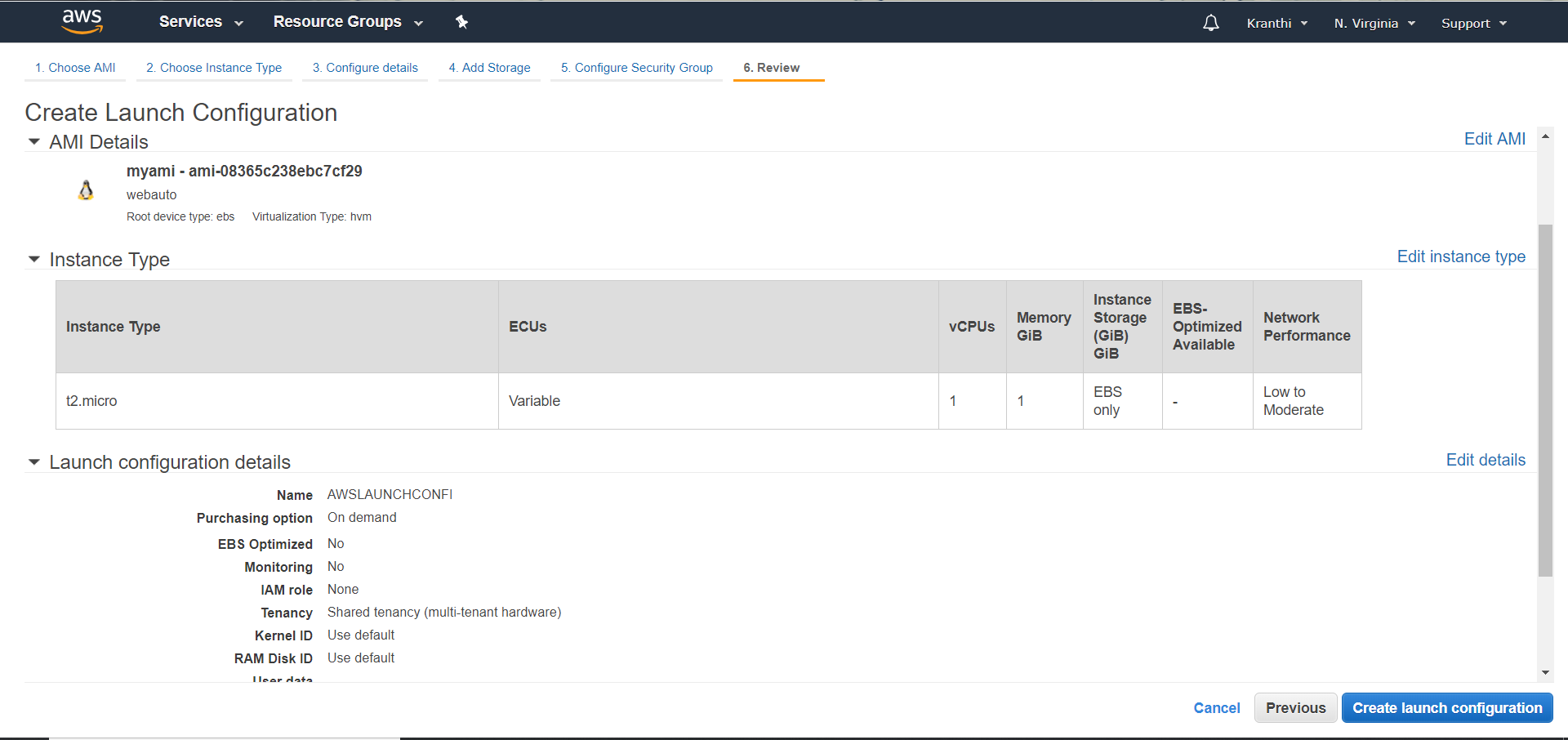
Storage settings to your instance.

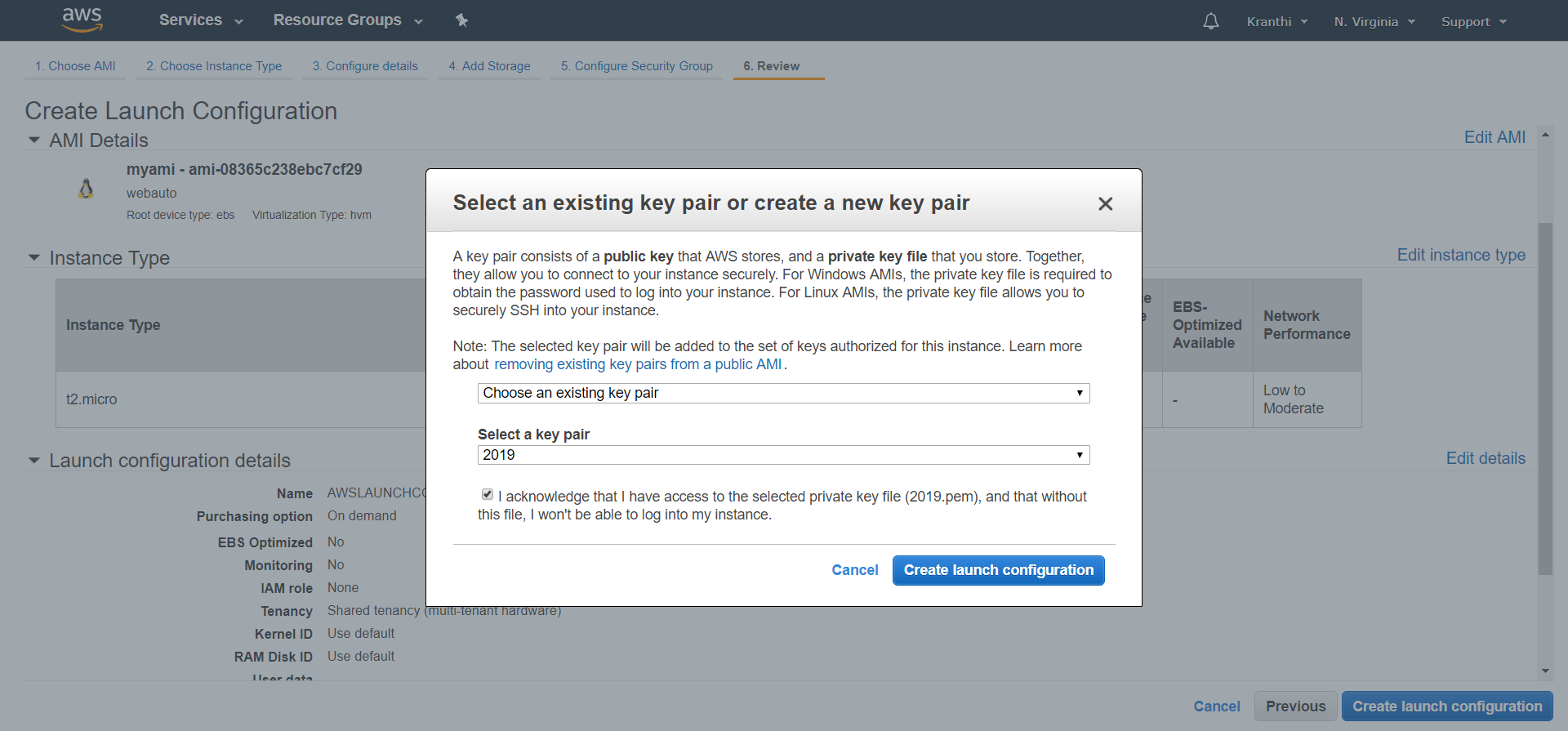
****

Configure security group and add own rules.

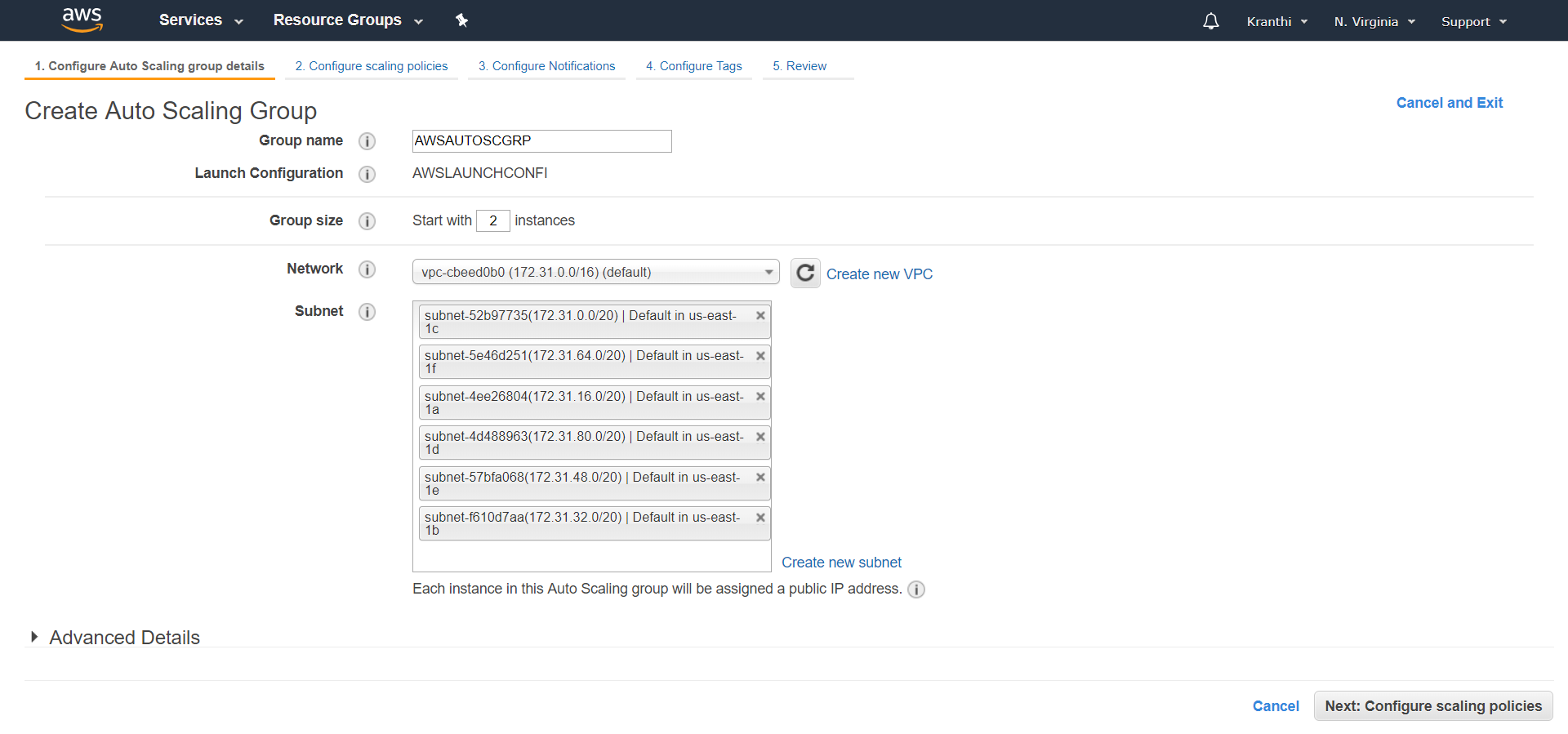
****

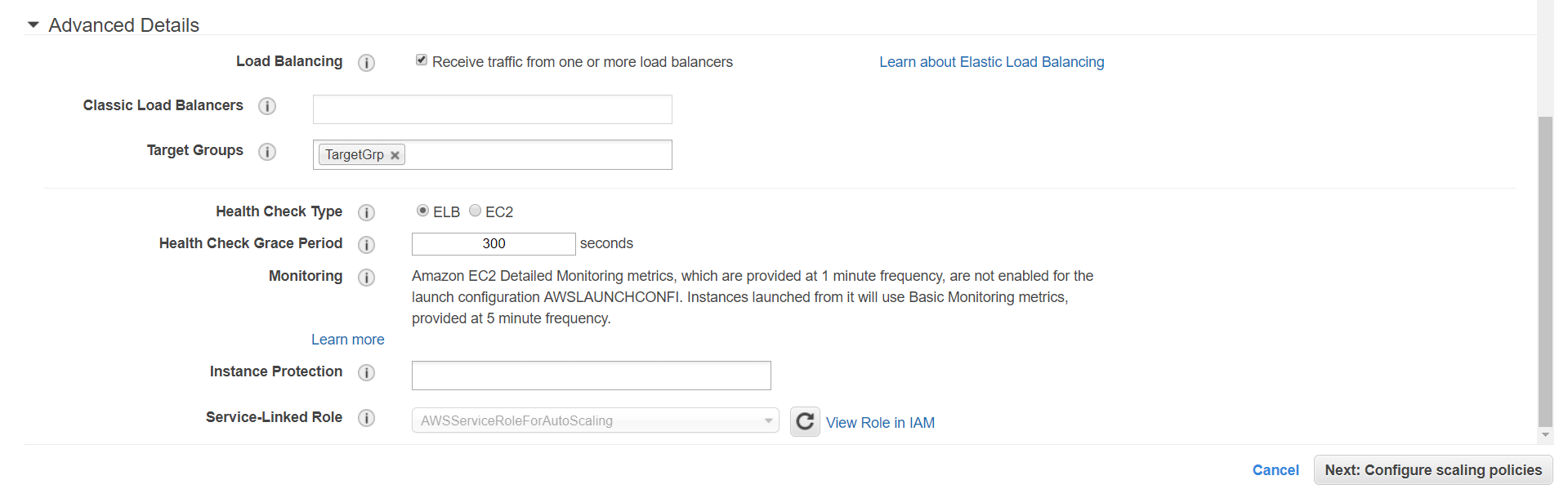
Review gives the total information about AMI.

****

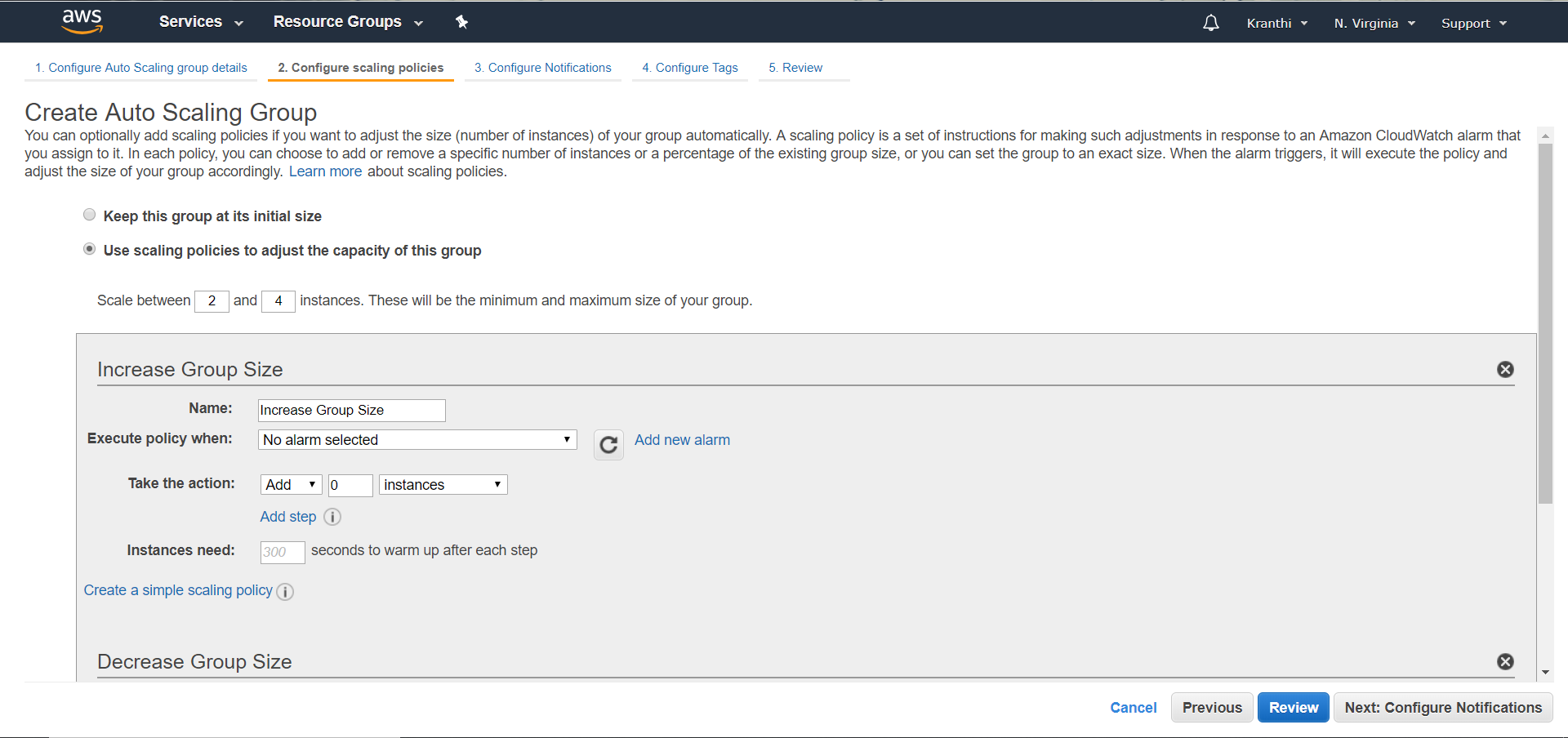
****

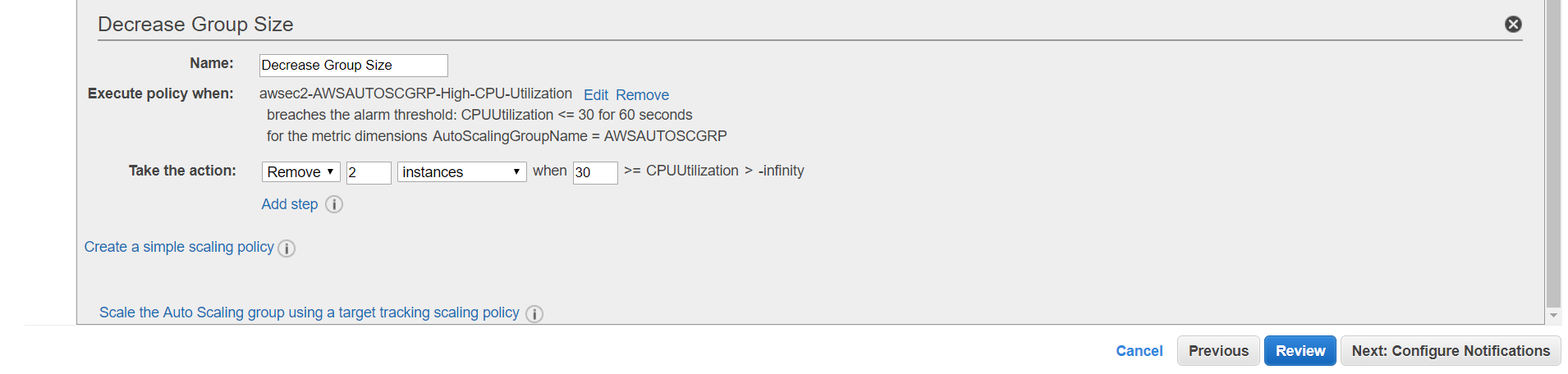
**9.3 Create Auto-Scaling groups after the launch configurations.**

****

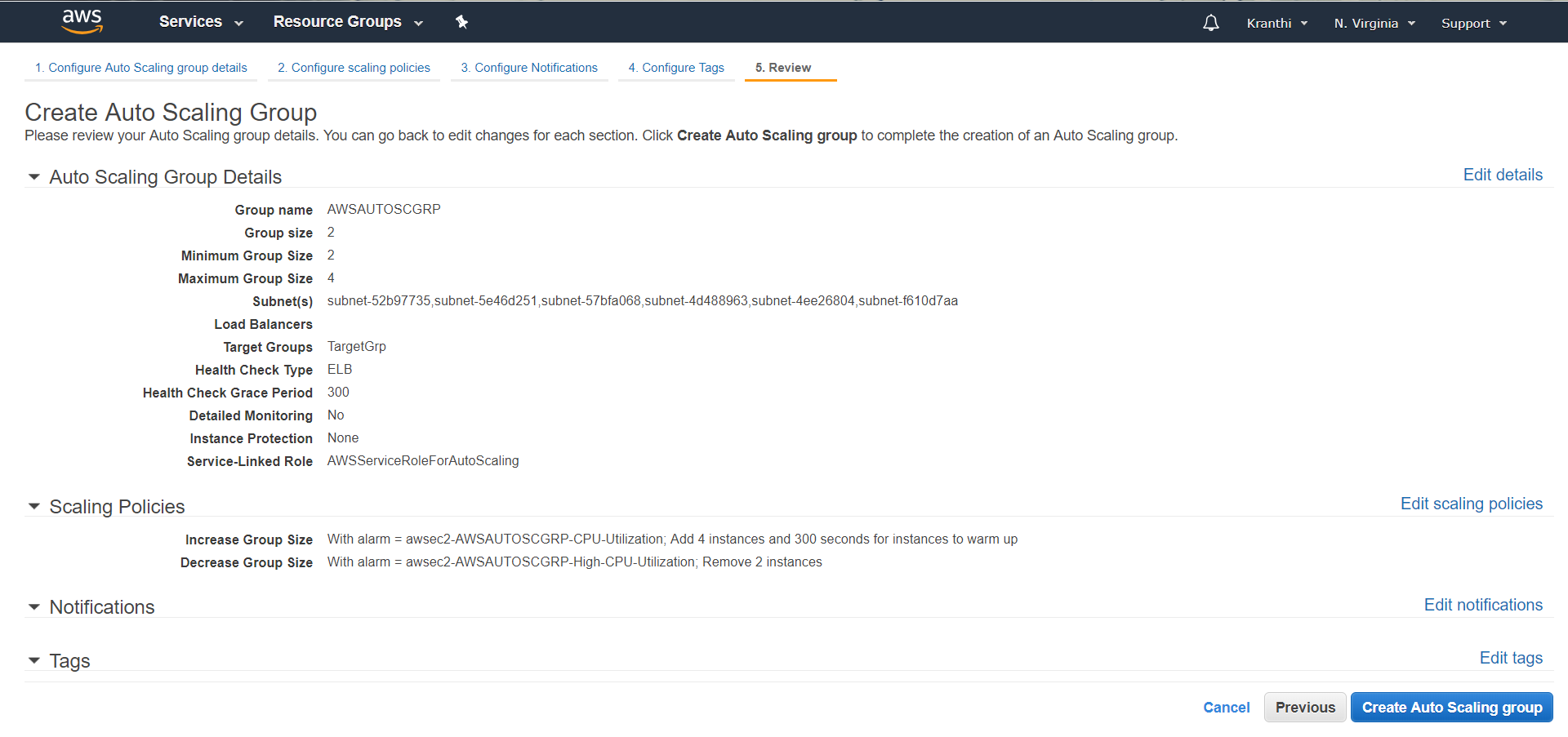
****

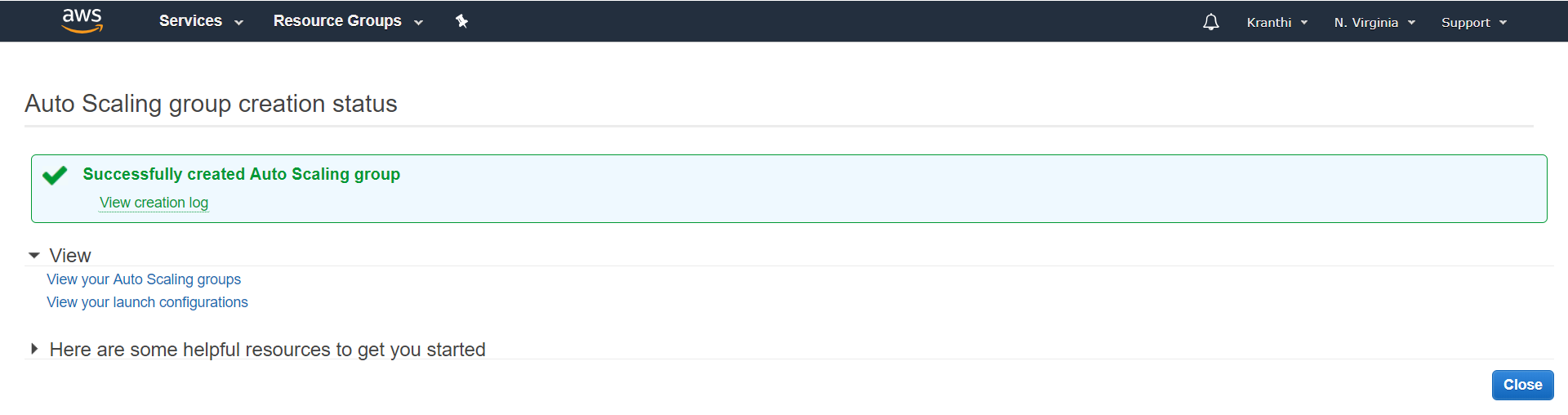
In this Auto-Scaling group, we need write how many instances we need to add when the avg-C.P.U percentages goes high or decrease.

****

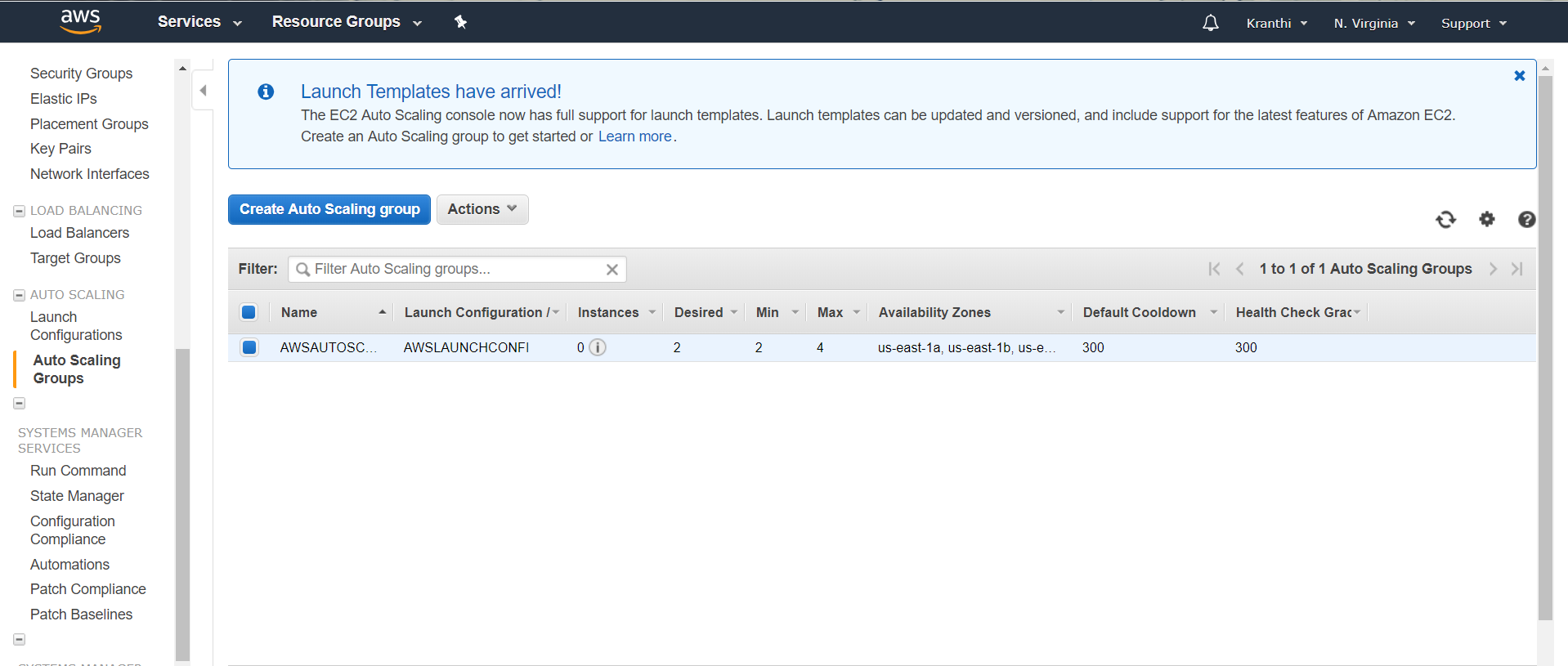
****

Reviews shows whatever we have added to the auto scaling group.

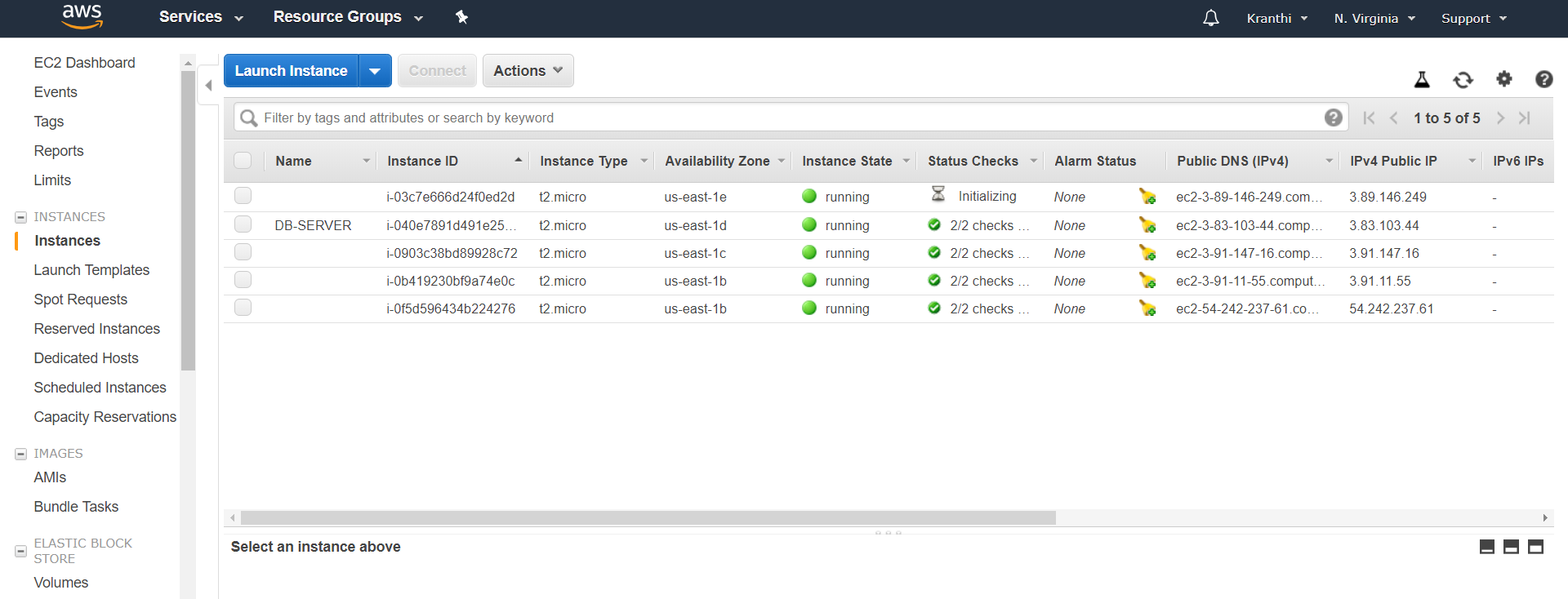
****

****

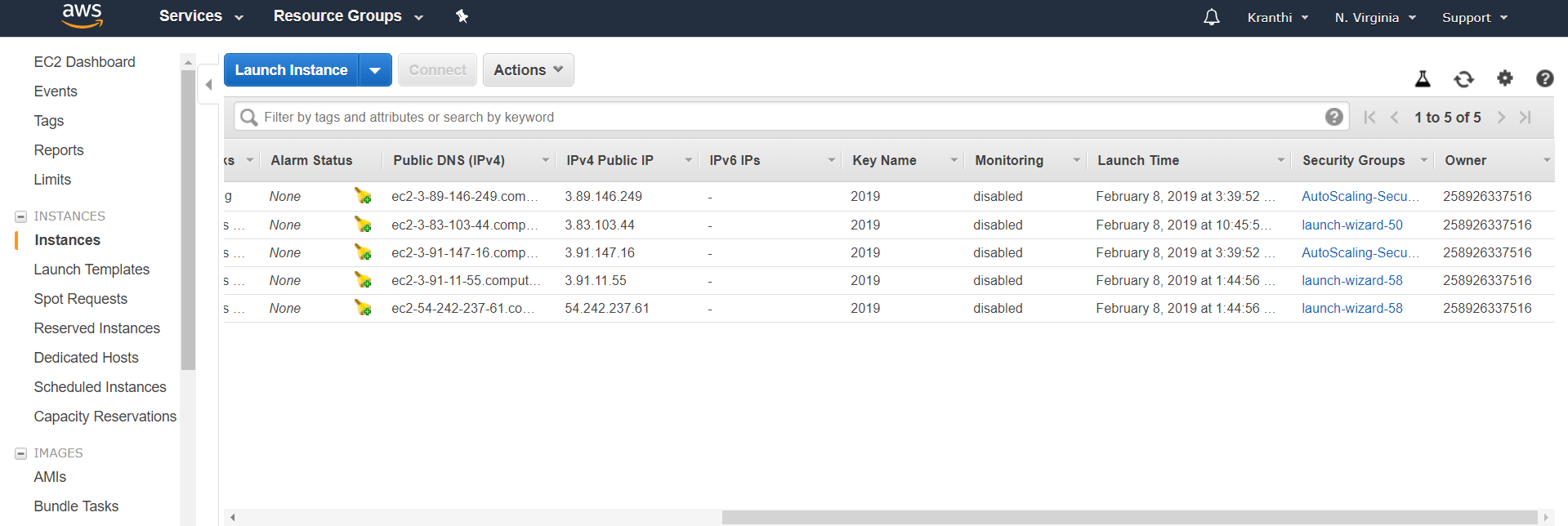
The complete adding of auto-scaling was finished and shows the added one.

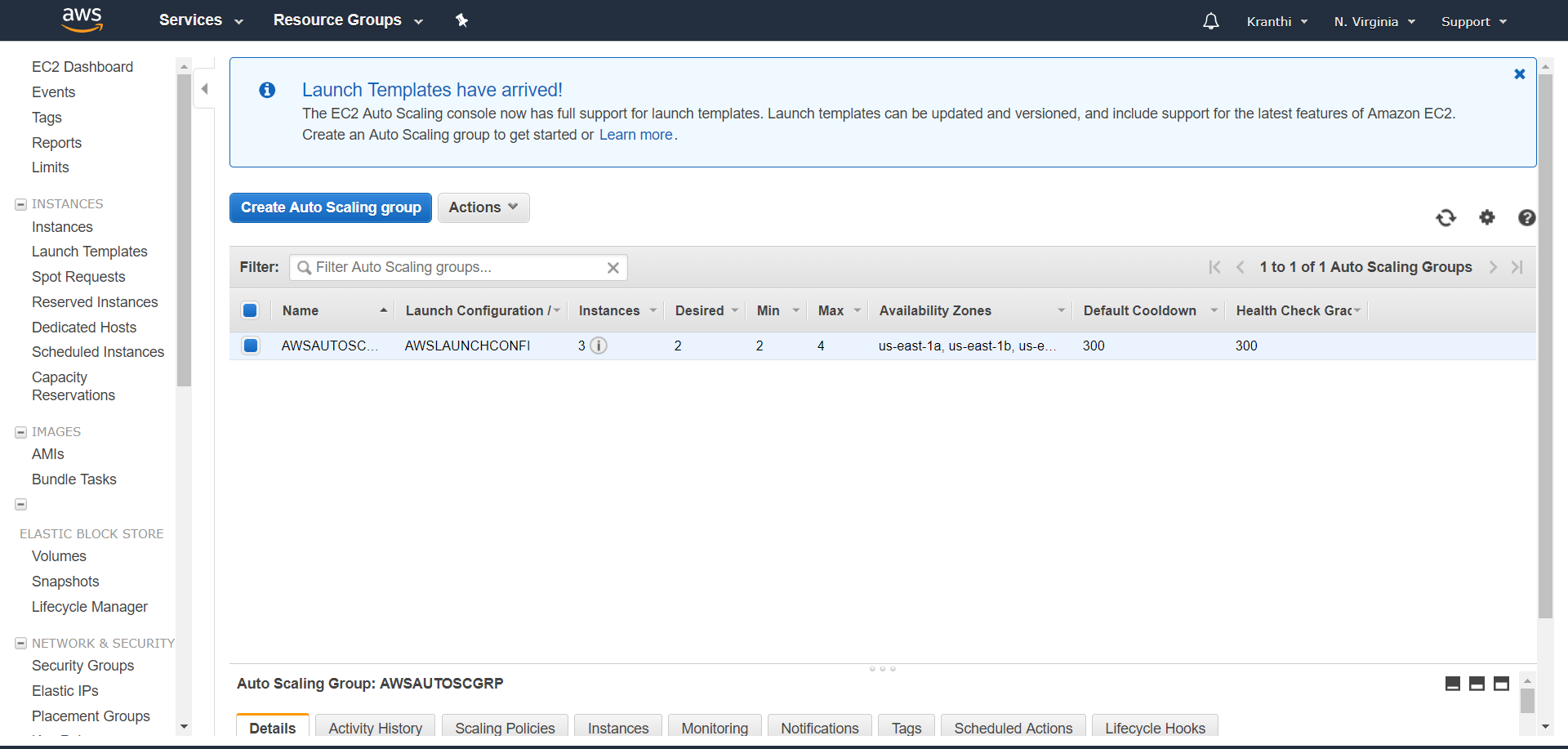
****

We can notice the automatic addition of new instances, when c.p.u percentage goes high.

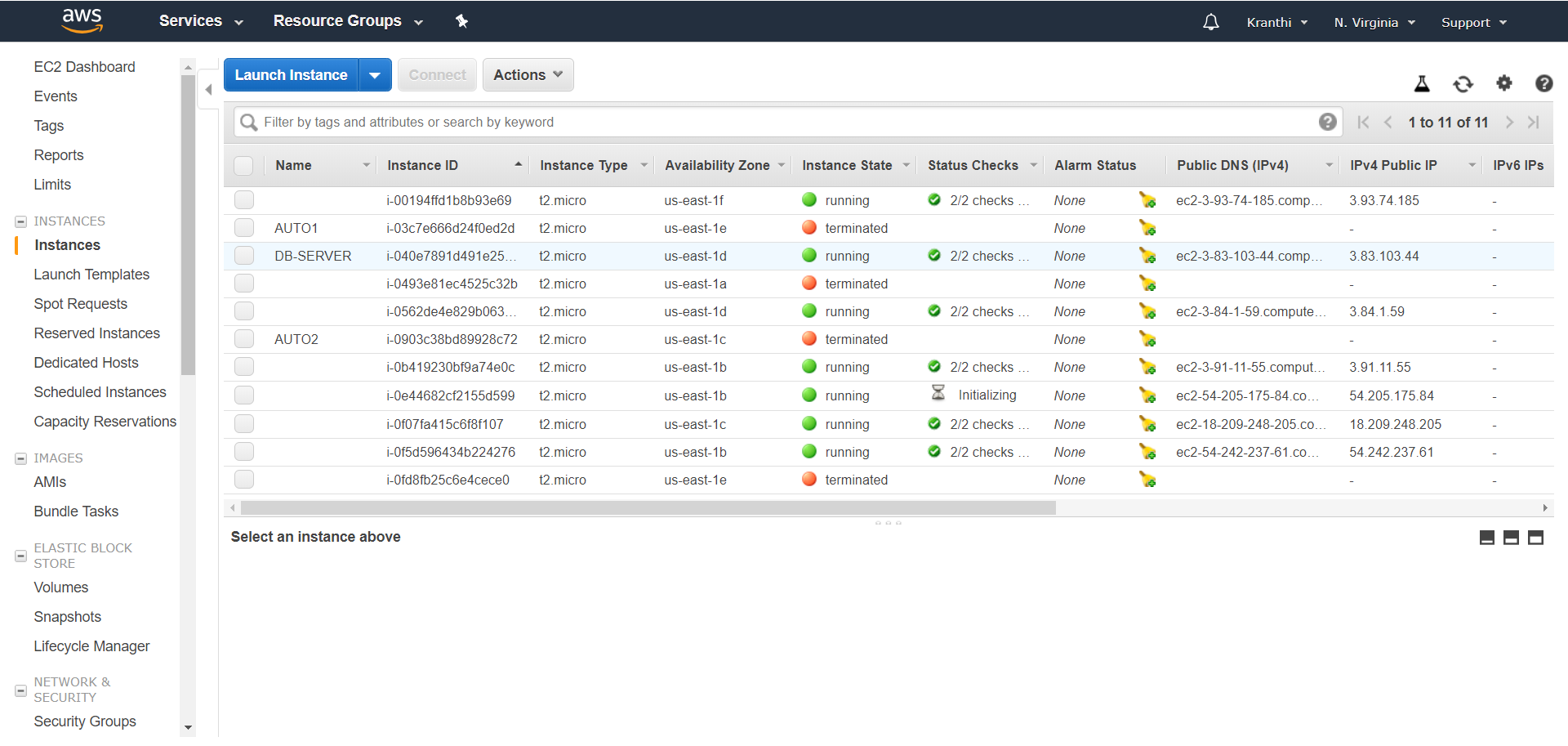
****

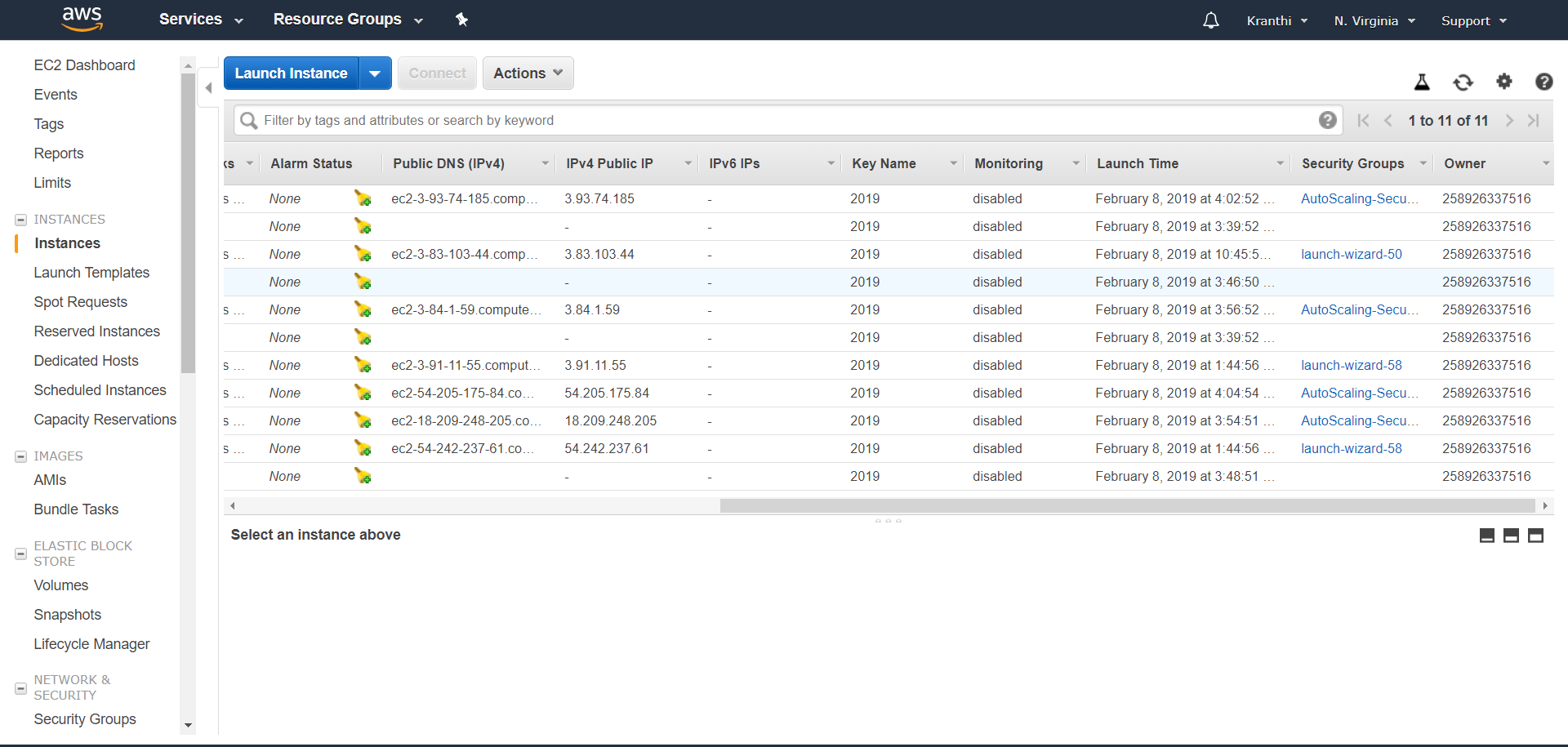
**Fig 9.1: Auto Scale in Output**

****

****

Termination of instances when the C.P.U percentage goes down.

**Fig 9.2: Auto Scale Out**

****

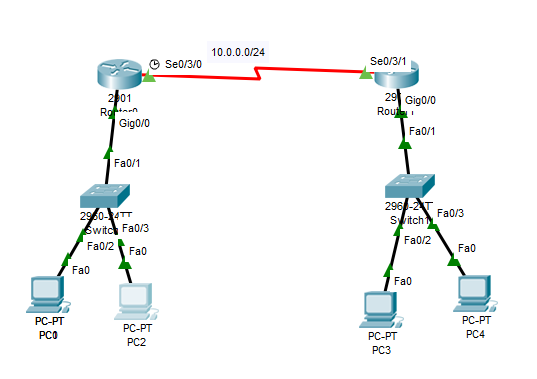
**Fig .9.3: Auto Scale Final Output**

**CHAPTER 10**

**NETWORK SECURITY**

**10.1 Introduction**

Network security consists of the policies and practices to prevent and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources. Only Network security can protect you from Trojan horse viruses. Network security involves the authorization of access to data in a system, controlled by the network administrator.



**Fig .10.1: Network Topology**

To avoid ARP SPOOFING, MAC FLOODING, and DHCP SPOOFING, we implement security policies. Firewall plays a preeminent role in network security. To prevent unauthorized access, we are using Cisco advanced security appliances. To connect different branches with security, we are implementing SITE TO SITE VPN. To overcome the network attacks, we are developing the Intrusion Prevention System. Cisco IOS Intrusion Prevention System (IPS) is an inline, deep-packet inspection feature that effectively mitigates a wide range of network attacks. These network security infrastructuresare implementing in On-premises, not in a cloud.

**10.2 DHCP Snooping**

DHCP snooping is a security feature that acts like a firewall between untrusted hosts and

Trusted DHCP Servers. The DHCP snooping feature performs the following activities:

* Validates DHCP messages received from untrusted sources and filters out invalid

Messages.

* Rate-limits DHCP traffic from trusted and untrusted sources.
* Builds and maintains the DHCP snooping binding database, which contains information about untrusted hosts with leased IP addresses.
* Utilizes the DHCP snooping binding database to validate subsequent requests from

Untrusted hosts

**Step by step procedure for implementing DHCP:**

The following steps are required to implement DHCP snooping on your network:

**Step 1.** Define and configure the DHCP server. Configuration of this step does not take

Place on the switch or router and is beyond the scope of this book.

**Step 2.** Enable DHCP snooping on at least one VLAN. By default, DHCP snooping is

Inactive on all VLANs.

**Step 3.** Ensure that DHCP server is connected through a trusted interface.

By default, the trust state of all interfaces is untrusted.

**Step 4.** Configure the DHCP snooping database agent. This step ensures that database

Entries are restored after a restart or switchover.

**Step 5.** Enable DHCP snooping globally

**Implementing DHCP on Switch:**

Enable DHCP Snooping Globally

sw2(config)# **ip dhcp snooping**

Enable DHCP Snooping on VLAN 10

sw2(config)# **ip dhcp snooping vlan 10**

Configure Interface Fa1/0/24 as a trusted interface

sw2(config)# **interface fa1/0/24**

sw2(config-if)# **ip dhcp snooping trust**

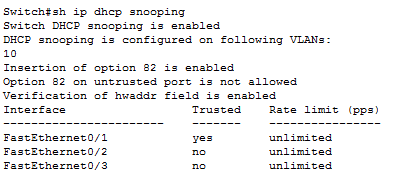
Configure the DHCP snooping database agent to store the bindings at a given location

sw2(config)# **exit**

sw2#

Verify DHCP Snooping Configuration

**Output:**

****

**Fig 10.2: DHCP Snooping Output**

**10.3 Port Security**

How many MAC addresses should legitimately show up inbound on an access port?

Port security controls how many MAC addresses can be learned on a single switch port.

This feature is implemented on a port-by-port basis. A typical user uses just a single MAC

Address.

Exceptions to this may be a virtual machine or two that might use different MAC

Addresses than their host, or if there is an IP phone with a built-in switch, which may also

Account for additional MAC addresses.

In any case, to avoid a user connecting dozens of devices to a switch that is then connected to their access port, you can use port security to limit the number of devices (MAC addresses) on each port.

This also protects against malicious applications that may be sending thousands of frames

Into the network, with a different bogus MAC address for each frame, as the user tries to

Exhaust the limits of the dynamic MAC address table on the switch, which might cause the

Switch to forward all frames to all ports within a VLAN so that the attacker can begin to

Sniff all packets. This is referred to as a *CAM table overflow attack*. *Content-addressable*

Memory *(CAM)* is a fancy way to refer to the MAC address table on the switch.

**Implementing Port-Security on Switch:**

SW2(config-if)# **interface fa 0/2**

! Enable the feature per interface

SW2(config-if)# **switchport port-security**

! Set the maximum to desired number. Default is 1. If we administratively

! set the maximum to 1, the command won't show in the running configuration

! because the configuration matches the default value. It is handy to know

! this behavior, so you won't be surprised by what may seem to be a missing

! part of your configuration.

SW2(config-if)# **switchport port-security maximum 5**

! Set the violation action. Default is err-disable. Protect will simply

! not allow

! frames from MAC addresses above the maximum.

SW2(config-if)# **switchport port-security violation protect**

This will cause the dynamic mac addresses to be placed into running

! -config to save them to startup config, use copy run start

SW2(config-if)# **switchport port-security mac-address sticky**

! To verify settings, use this command

**Output:**

Switch#sh port-security

Secure Port MaxSecure Addr CurrentAddr Security Violation Security Action

(Count) (Count) (Count)

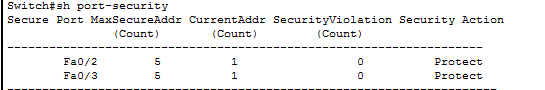
--------------------------------------------------------------------

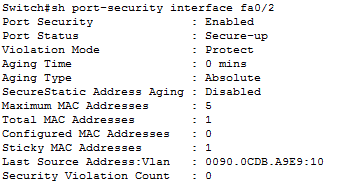
Fa0/2 5 1 0 Protect

Fa0/3 5 1 0 Protect

----------------------------------------------------------------------

Switch#





**Fig 10.3: Port Security Output**

**10.4 Securing the Cisco IOS Image and Configuration Files**

If a router has been compromised, and the flash file system and NVRAM have been deleted,

Then there could be significant downtime as the files are put back in place before restoring normal router functionality. The Cisco Resilient Configuration feature is intended to improve the recovery time by making a secure working copy of the IOS image and start up configuration files (which are referred to as the *primary* boot set) that cannot be deleted by a remote user.

To enable and save the primary boot set to a secure archive in persistent storage, follow Secure the IOS image

R6(config)# **secure boot-image**

%IOS\_RESILIENCE-5-IMAGE\_RESIL\_ACTIVE: Successfully secured running image

! Secure the startup-config

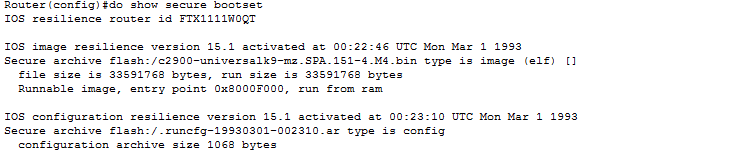
R6(config)# **secure boot-config**

%IOS\_RESILIENCE-5-CONFIG\_RESIL\_ACTIVE: Successfully secured config archive

[flash:.runcfg-20111222-230018.ar]

! Verify the bootset

**Output:**



**Fig 10.5: Secured IOS Output**

**10.5 ARP Dynamic Inspection**

ARP provides IP communication within a Layer 2 broadcast domain by mapping an IP

Address to a MAC address. ARP spoofing attacks and ARP cache poisoning can occur because ARP allows a gratuitous reply from a host even if an ARP request was not received. After the attack, all traffic from the device under attack flows through the attacker’s computer and then to the router, switch, host.

An ARP spoofing attack can target hosts, switches, and routers connected to your Layer

2 network by poisoning the ARP caches of systems connected to the subnet and by intercepting traffic intended for other hosts on the subnet.

DAI is a security feature that validates ARP packets in a network. DAI intercepts, logs, and

Discards ARP packets with invalid IP-t o -MAC address bindings. This capability protects the

Network from some man-in-the-middle attacks.

DAI determines the validity of an ARP packet based on valid IP-to-MAC address bindings

Stored in a trusted database, the DHCP snooping binding database. As described in the previous section, this database is built by DHCP snooping if DHCP snooping is enabled on the VLANs and on the switch. If the ARP packet is received on a trusted interface, the switch

Forward the packet without any checks. On untrusted interfaces, the switch forwards the packet only if it is valid.

Enable DAI on VLAN 10

sw2(config)# **ip arp inspection vlan 10**

sw2(config)# **exit**

! Verify DAI Configuration for VLAN 10

sw2# **show ip arp inspection vlan 10**

Source Mac Validation : Disabled

Destination Mac Validation : Disabled

IP Address Validation : Disabled

Vlan Configuration Operation ACL Match Static ACL

---- ------------- --------- --------- ----------

10 Enabled Inactive

Vlan ACL Logging DHCP Logging Probe Logging

---- ----------- ------------ -------------

10 Deny Deny Off

! Configure Interface Fa1/0/24 as a Trusted DAI Interface

sw2(config)# **interface fa1/0/24**

sw2(config-if)# **ip arp inspection trust**

sw2(config-if)# **exit**

sw2(config)# **exit**

sw2# **show ip arp inspection interfaces**

Interface Trust State Rate (pps) Burst Interval

--------------- ----------- ---------- --------------

Fa1/0/1 Untrusted 15 1

Fa1/0/2 Untrusted 15 1

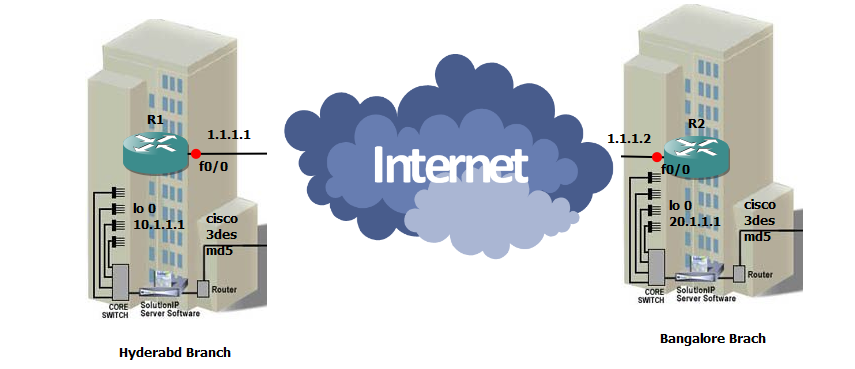
! output removed for brevity

Fa1/0/23 Untrusted 15 1

Fa1/0/24 Trusted None N/A

**.**

**10.6 Site to Site VPN**

****

**Fig 10.6: Site to Site VPN**

It helps to connect two different branches or remote locations. It ensures that connectivity between two public networks and with security.

**Implementation Stages:**

Phase 1: Internet security association key management protocols (ISAKMP).

Phase 2: Internet protocol security. It is having ESP and AH.

Phase 3: Interesting traffic will be configured by ACL.

Phase 4: Mapping (Crypto Map).

Phase 5: Apply map on interface.

Here Pre shared key will be used for trust worthiness between two branches.

**Encryption:**

It will convert plain text into cipher text. It classified into 3 types.

1) DES-Data encryption standard

2) 3 DES

3) AES-Advanced encryption standard

**Hash:** It is used for checksum. It is having two types

1) MD5-Message Digest

2) SHA-Secure hashing Algorithm

**Ockley:** It is responsible to carry the message from Router 1 to Router 2.

If we are using both sites of routers are cisco vendors, then it is operated in main mode, in this 6 messages will be exchanged. For different vendors of router in both sites, it will be operated in Aggressive Mode only 3 messages exchanged.

In this VPN, we have 2 Tunnels. Those are

1) ISAKMP

2) IPSEC

**ISAKMP:** By using this tunnel session keys will be exchanged, this was generated by the **Diffie**–**Hellman** Algorithm. ISAKMP tunnel is responsible for IPSEC tunnel up and running.

**IPSEC:** This tunnel is used for actual data transmission. ESP and AH are responsible to carry data in IPSEC tunnel.

**Crypto Map:** It is used to identify the router the packet belongs to which network. Only one crypto map can be applied for single interface.

**HMac:** Hmac is used for sequence number and to add the Tags.

**10.6.1 VPN CONFIGURATION:**

**Router-1**

crypto isakmp policy 10

encr 3des

hash md5

authentication pre-share

group 2

crypto isakmp key cisco address 1.1.1.2

crypto isakmp key juniper address 1.1.1.3

!

!

crypto ipsec transform-set dell esp-3des esp-md5-hmac

crypto ipsec transform-set lenovo esp-aes esp-sha-hmac

!

crypto map irfan 110 ipsec-isakmp

set peer 1.1.1.2

set transform-set dell

match address 101

crypto map irfan 111 ipsec-isakmp

set peer 1.1.1.3

set transform-set lenovo

match address 102

!

interface Loopback0

ip address 10.1.1.1 255.0.0.0

!

interface FastEthernet0/0

ip address 1.1.1.1 255.0.0.0

duplex auto

speed auto

crypto map irfan

!

interface FastEthernet0/1

no ip address

shutdown

duplex auto

speed auto

!

ip route 0.0.0.0 0.0.0.0 FastEthernet0/0

!

no ip http server

no ip http secure-server

!

access-list 101 permit ip host 10.1.1.1 host 20.1.1.1

access-list 101 permit ip host 1.1.1.1 host 1.1.1.2

access-list 102 permit ip host 10.1.1.1 host 30.1.1.1

!

control-plane

!

line con 0

exec-timeout 0 0

privilege level 15

logging synchronous

line aux 0

exec-timeout 0 0

privilege level 15

logging synchronous

line vty 0 4

login

!

!

End

**Router-2**

crypto isakmp policy 10

encr 3des

hash md5

authentication pre-share

group 2

crypto isakmp key cisco address 1.1.1.1

!

!

crypto ipsec transform-set dell esp-3des esp-md5-hmac

!

crypto map irfan 110 ipsec-isakmp

set peer 1.1.1.1

set transform-set dell

match address 101

!

!

!

!

!

interface Loopback0

ip address 20.1.1.1 255.0.0.0

!

interface FastEthernet0/0

ip address 1.1.1.2 255.0.0.0

duplex auto

speed auto

crypto map irfan

!

interface FastEthernet0/1

no ip address

shutdown

duplex auto

speed auto

!

ip route 0.0.0.0 0.0.0.0 1.1.1.1

!

no ip http server

no ip http secure-server

!

access-list 101 permit ip host 20.1.1.1 host 10.1.1.1

access-list 101 permit ip host 1.1.1.2 host 1.1.1.1

!

!

control-planel!

!

line con 0

exec-timeout 0 0

privilege level 15

logging synchronous

line aux 0

exec-timeout 0 0

privilege level 15

logging synchronous

line vty 0 4

login

End

**OUTPUT**

R1#sh crypto isakmp sa

dst src state conn-id slot status

1.1.1.2 1.1.1.1 QM\_IDLE 1 0 ACTIVE

R1#sh crypto ipsec sa

interface: FastEthernet0/0

Crypto map tag: irfan, local addr 1.1.1.1

protected vrf: (none)

local ident (addr/mask/prot/port): (1.1.1.1/255.255.255.255/0/0)

remote ident (addr/mask/prot/port): (1.1.1.2/255.255.255.255/0/0)

current\_peer 1.1.1.2 port 500

PERMIT, flags={origin\_is\_acl,}

#pkts encaps: 0, #pkts encrypt: 0, #pkts digest: 0

#pkts decaps: 0, #pkts decrypt: 0, #pkts verify: 0

#pkts compressed: 0, #pkts decompressed: 0

#pkts not compressed: 0, #pkts compr. failed: 0

#pkts not decompressed: 0, #pkts decompress failed: 0

#send errors 0, #recv errors 0

local crypto endpt.: 1.1.1.1, remote crypto endpt.: 1.1.1.2

path mtu 1500, ip mtu 1500

current outbound spi: 0x0(0)

inbound esp sas:

inbound ah sas:

inbound pcp sas:

outbound esp sas:

outbound ah sas:

outbound pcp sas:

protected vrf: (none)

local ident (addr/mask/prot/port): (10.1.1.1/255.255.255.255/0/0)

remote ident (addr/mask/prot/port): (20.1.1.1/255.255.255.255/0/0)

current\_peer 1.1.1.2 port 500

PERMIT, flags={origin\_is\_acl,}

#pkts encaps: 14, #pkts encrypt: 14, #pkts digest: 14

#pkts decaps: 14, #pkts decrypt: 14, #pkts verify: 14

#pkts compressed: 0, #pkts decompressed: 0

#pkts not compressed: 0, #pkts compr. failed: 0

#pkts not decompressed: 0, #pkts decompress failed: 0

#send errors 6, #recv errors 0

local crypto endpt.: 1.1.1.1, remote crypto endpt.: 1.1.1.2

path mtu 1500, ip mtu 1500

current outbound spi: 0x65DCEB8B(1708977035)

inbound esp sas:

spi: 0xDA81E22F(3665945135)

transform: esp-3des esp-md5-hmac ,

in use settings ={Tunnel, }

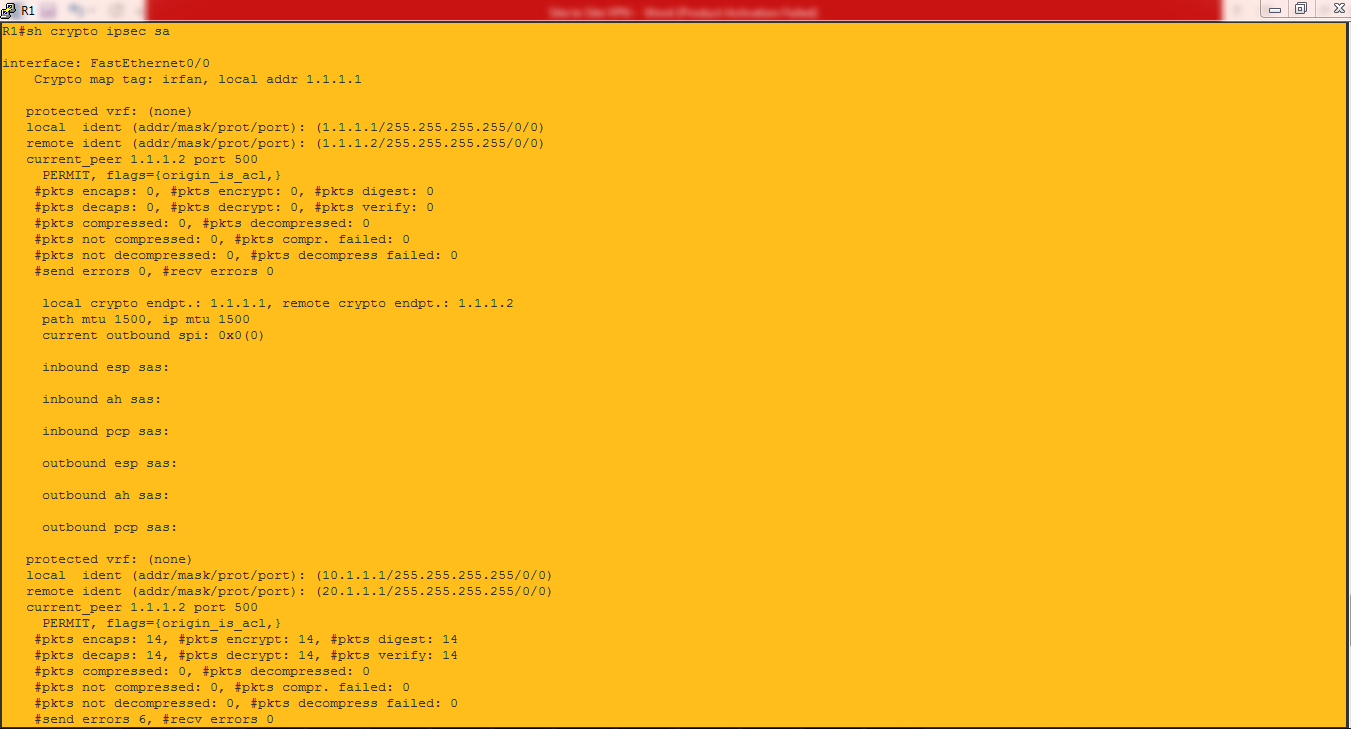
conn id: 2004, flow\_id: SW:4, crypto map: irfan

sa timing: remaining key lifetime (k/sec): (4441307/3533)

IV size: 8 bytes

replay detection support: Y

Status: ACTIVE



**Fig 10.6.1: VPN Output**

**CHAPTER 11**

**CONCLUSION & FUTURE SCOPES:**

The main theme of our project is to maintain the web applications run without any down time even when the number of users using the application is more may be in billions. AWS provides auto scaling that is, addition of new servers that maintains the same web application details when the C.P.U percentage gets increased and deleting the newly created servers when the usage is normal and load balancing features, By using these services provided by AWS, it costs less while compared to maintaining the physical servers, payable as per our usage, monitored 24/7 without any down time.

Using IOT sensors we can get the information regarding ATMOSPHERE such as Temperature, Humidity, Raining Status.

Finally, there is a clear need to better understand and design networked systems that are both robust to variations in the components and secured networks. We have provided some security features in On-Premises such as PORT-SECURITY, DHCP Snooping, SITE-To-SITE-VPN, Securing Cisco IOS and Configurations. Leaving the default configurations may cause cyber-attacks so preventing our infrastructure from these attacks proves to be a well secured network.

**FUTURE SCOPES:**

Our project can be very useful for those who are new to this digital world, who wants to establish a new e-commerce website and wants to deploy every possible security for the website including load balancing and Auto scaling. Beside this it’s very useful for the people who don’t know about MySQL or database its security and implementation of these.

Hence its will be surely be recognized one day as a main step for developing website for an individual and throughout IOT project we can also make our website connected with IOT devices.

Developing big e-commerce website or any multimedia streaming website or any web related project can be made by just deploying with better security for any individual. It could also be used in Colleges, Universities or any other sector.

Eg: DB Migration & IPS

**CHAPTER 12**

**REFERENCES**

https://searchcloudcomputing.techtarget.com/definition/Software-as-a-Service

https://searchcloudcomputing.techtarget.com/definition/Platform-as-a-Service-PaaS

https://searchcloudcomputing.techtarget.com/definition/Infrastructure-as-a-Service-IaaS

AWS DOCUMENTATION from AWS website.

CISCO DOCUMENTATION form cisco website

https://hackernoon.com/tutorial-creating-and-managing-a-node-js-server-on-aws-part-1-d67367ac5171

https://websiteforstudents.com/install-the-latest-node-js-and-nmp-packages-on-ubuntu-16-04-18-04-lts/

https://stackoverflow.com/questions/17666249/how-to-import-an-sql-file-using-the-command-line-in-mysql

**APPENDIX**

**Html code for our website**

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>Commercial</title>

<meta content="width=device-width, initial-scale=1.0" name="viewport">

<meta content="" name="keywords">

<meta content="" name="description">

<!-- Favicons -->

<link href="img/img1.jpg" rel="icon">

<!-- Google Fonts -->

<link href="https://fonts.googleapis.com/css?family=Open+Sans:300,400,400i,600,700|Raleway:300,400,400i,500,500i,700,800,900" rel="stylesheet">

<!-- Bootstrap CSS File -->

<link href="lib/bootstrap/css/bootstrap.min.css" rel="stylesheet">

<!-- Libraries CSS Files -->

<link href="lib/nivo-slider/css/nivo-slider.css" rel="stylesheet">

<link href="lib/owlcarousel/owl.carousel.css" rel="stylesheet">

<link href="lib/owlcarousel/owl.transitions.css" rel="stylesheet">

<link href="lib/font-awesome/css/font-awesome.min.css" rel="stylesheet">

<link href="lib/animate/animate.min.css" rel="stylesheet">

<link href="lib/venobox/venobox.css" rel="stylesheet">

<!-- Nivo Slider Theme -->

<link href="css/nivo-slider-theme.css" rel="stylesheet">

<!-- Main Stylesheet File -->

<link href="css/style.css" rel="stylesheet">

<!-- Responsive Stylesheet File -->

<link href="css/responsive.css" rel="stylesheet">

<style>

.grid-container {

display: grid;

grid-template-columns: auto auto auto auto;

grid-gap: 10px;

background-color:hsl(0, 100%, 90%);;

padding: 10px;

}

}

</style>

</head>

<body data-spy="scroll" data-target="#navbar-example">

<div id="preloader"></div>

<header>

<!-- header-area start -->

<div id="sticker" class="header-area">

<div class="container">

<div class="row">

<div class="col-md-12 col-sm-12">

<!-- Navigation -->

<nav class="navbar navbar-default">

<!-- Brand and toggle get grouped for better mobile display →

<div class="navbar-header">

<button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target=".bs-example-navbar-collapse-1" aria-expanded="false">

<span class="sr-only">Toggle navigation</span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

<!-- Brand -->

<a class="navbar-brand page-scroll sticky-logo" href="index.html">

<h1>Flipkart</h1>

<!-- Uncomment below if you prefer to use an image logo -->

<!-- <img src="img/logo.png" alt="" title=""> -->

</a>

</div>

<!-- Collect the nav links, forms, and other content for toggling -->

<div class="collapse navbar-collapse main-menu bs-example-navbar-collapse-1" id="navbar-example">

<ul class="nav navbar-nav navbar-right">

<li class="active">

<a class="page-scroll" href="#home">Home</a>

</li>

<li>

<a class="page-scroll" href="#about">Sales</a>

</li>

<li>

<a class="page-scroll" href="iot.html">IOT</a>

</li>

<li>

<a class="page-scroll">Dashboard</a>

</li>

<li>

<a class="page-scroll">Cart</a>

</li>

</ul>

</div>

<!-- navbar-collapse -->

</nav>

<!-- END: Navigation -->

</div>

</div>

</div>

</div>

<!-- header-area end -->

</header>

<!-- header end -->

<!-- Start Slider Area -->

<div id="home" class="slider-area">

<div class="bend niceties preview-2">

<div id="ensign-nivoslider" class="slides">

<img src="img/slider/img3.jpg" alt="" title="#slider-direction-1" />

<img src="img/slider/img4.png" alt="" title="#slider-direction-2" />

<img src="img/slider/img5.png" alt="" title="#slider-direction-3" />

</div>

<!-- End Slider Area -->

</div>

</div>

<!-- direction 1 -->

<div id="slider-direction-1" class="slider-direction slider-one">

<div class="container">

<div class="row">

<div class="col-md-12 col-sm-12 col-xs-12">

<div class="slider-content">

<!-- layer 1 -->

<div class="layer-1-1 hidden-xs wow slideInDown" data-wow-duration="2s" data-wow-delay=".2s">

<h2 class="title1" style="color:#FF3855; text-shadow:1px 1px 1px grey;">The Best Sales </h2>

</div>

<!-- layer 2 -->

<div class="layer-1-3 hidden-xs wow slideInUp" data-wow-duration="2s" data-wow-delay=".2s">

<a class="ready-btn right-btn page-scroll" href="#services">See Offers</a>

<a class="ready-btn page-scroll" href="#about">Buy More</a>

</div>

</div>

</div>

</div>

</div>

</div><!-- direction 2 -->

<div id="slider-direction-2" class="slider-direction slider-two">

<div class="container">

<div class="row">

<div class="col-md-12 col-sm-12 col-xs-12">

<div class="slider-content">

<!-- layer 1 -->

<div class="layer-1-1 hidden-xs wow slideInDown" data-wow-duration="2s" data-wow-delay=".2s">

<h2 class="title1" style="color:#1B1B1B; text-shadow:1px 1px 1px grey;">The Best Sales and Offers</h2>

</div>

<!-- layer 2 -->

<div class="layer-1-2 wow slideInUp" data-wow-duration="2s" data-wow-delay=".1s">

<h1 class="title2" style="color:#1B1B1B; text-shadow:1px 1px 1px grey;">Hurray It's a Big Billion Day</h1>

</div>

<!-- layer 3 -->

<div class="layer-1-3 hidden-xs wow slideInUp" data-wow-duration="2s" data-wow-delay=".2s">

<a class="ready-btn right-btn page-scroll" href="#services">See Offers</a>

<a class="ready-btn page-scroll" href="#about">Buy More</a>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- direction 1 -->

<div id="slider-direction-3" class="slider-direction slider-three">

<div class="container">

<div class="row">

<div class="col-md-12 col-sm-12 col-xs-12">

<div class="slider-content">

<!-- layer 1 -->

<div class="layer-1-1 hidden-xs wow slideInDown" data-wow-duration="2s" data-wow-delay=".2s">

<h2 class="title1" style="color:#1B1B1B;">Great Indian Sales</h2>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- Start sales Area -->

<div class="grid-container">

<div class="container text-center" >

<div class="row">

<h2>Deals of the day</h2>

<!--<button id="btn\_click">view more</button!-->

</div>

</div>

</div>

<div class="row">

<div class="col-md-4 col-sm-3 col-xs-12">

<div class="single-team-member">

<a href="#">

<img src="img/sales/14.jpg" alt="">

</a>

<div class="team-social-icon text-center">

<ul>

<li>

<a href="#">

<i class="fa fa-gittip"></i>

</a>

</li>

</ul>

</div>

</div>

<div class="team-content text-center">

<h4>Technology in next level</h4>

<p>Offer today</p>

<p>₹5400 48%off</p>

</div>

</div>

<!-- End column -->

<!-- <div class="team-top"> -->

<div class="col-md-4 col-sm-3 col-xs-12">

<div class="single-team-member">

<a href="#">

<img src="img/sales/12.jpg" alt="">

</a>

<div class="team-social-icon text-center">

<ul>

<li>

<a href="#">

<i class="fa fa-gittip"></i>

</a>

</li>

</ul>

</div>

</div>

<div class="team-content text-center">

<h4>shoes & more</h4>

<p>₹690 60% off</p>

</div>

</div>

<!-- End column -->

<!-- <div class="team-top"> -->

<div class="col-md-4 col-sm-3 col-xs-12">

<div class="single-team-member">

<a href="#">

<img src="img/sales/13.jpg" alt="">

</a>

<div class="team-social-icon text-center">

<ul>

<li>

<a href="#">

<i class="fa fa-gittip"></i>

</a>

</li>

</ul>

</div>

</div>

<div class="team-content text-center">

<h4>Girls Fashion wear</h4>

<p>₹250 74% off</p>

</div>

</div>

</div>

<!-- End column -->

</div>

</div>

</div>

</div>

<!-- End Team Area -->

<!-- Start About area -->

<div id="about" class="about-area area-padding">

<div class="container">

<div class="row">

<div class="col-md-12 col-sm-12 col-xs-12">

<div class="section-headline text-center">

<h2>Great Sales and Exclusive Offers</h2>

</div>

</div>

</div>

<div class="row">

<!-- Start Portfolio -page -->

<div class="awesome-project-1 fix">

<div class="col-lg-12 col-md-12 col-sm-12 col-xs-12">

<div class="awesome-menu ">

<ul class="project-menu">

<li>

<a href="#" class="active" data-filter="\*">All</a>

</li>

<li>

<a href="#" data-filter=".design">Acsessories</a>

</li>

<li>

<a href="#" data-filter=".photo">electronics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="awesome-project-content">

<!-- single-awesome-project start -->

<div class="col-md-4 col-sm-4 col-xs-12 design development">

<div class="single-awesome-project">

<div class="awesome-img">

<a href="#"><img src="img/sales/kitchen.jpg" alt="" /></a>

<div class="add-actions text-center">

<div class="project-dec">

<a class="venobox" data-gall="myGallery" href="img/sales/kitchen.jpg">

<h4>Kitchen Items 50% off</h4>

</a>

</div>

</div>

</div>

</div>

</div>

<!-- single-awesome-project end -->

<!-- single-awesome-project start -->

<div class="col-md-4 col-sm-4 col-xs-12 design">

<div class="single-awesome-project">

<div class="awesome-img">

<a href="#"><img src="img/sales/winter.jpg" alt="" /></a>

<div class="add-actions text-center">

<div class="project-dec">

<a class="venobox" data-gall="myGallery" href="img/sales/winter.jpg">

<h4>Winter shopping</h4>

</a>

</div>

</div>

</div>

</div>

</div>

<!-- single-awesome-project end -->

<!-- single-awesome-project start -->

<div class="col-md-4 col-sm-4 col-xs-12 photo development">

<div class="single-awesome-project">

<div class="awesome-img">

<a href="#"><img src="img/sales/accessories.jpg" alt="" /></a>

<div class="add-actions text-center">

<div class="project-dec">

<a class="venobox" data-gall="myGallery" href="img/sales/accessories.jpg">

<h4>Gadgets</h4>

<span>On sales</span>

</a>

</div>

</div>

</div>

</div>

</div>

<!-- single-awesome-project end -->

<!-- single-awesome-project start -->

<div class="col-md-4 col-sm-4 col-xs-12 design">

<div class="single-awesome-project">

<div class="awesome-img">

<a href="#"><img src="img/sales/7.jpeg" alt="" /></a>

<div class="add-actions text-center">

<div class="project-dec">

<a class="venobox" data-gall="myGallery" href="img/sales/7.jpeg">

<h4>Fashion Week</h4>

<span>40% sales</span>

</a>

</div>

</div>

</div>

</div>

</div>

<!-- single-awesome-project end -->

<!-- single-awesome-project start -->

<div class="col-md-4 col-sm-4 col-xs-12 photo development">

<div class="single-awesome-project">

<div class="awesome-img">

<a href="#"><img src="img/sales/galaxy.jpg" alt="" /></a>

<div class="add-actions text-center">

<div class="project-dec">

<a class="venobox" data-gall="myGallery" href="img/sales/galaxy.jpg">

<h4>Galaxy</h4>

<span>On sales</span>

</a>

</div>

</div>

</div>

</div>

</div>

<!-- single-awesome-project end -->

<div class="col-md-4 col-sm-4 col-xs-12 design">

<div class="single-awesome-project">

<div class="awesome-img">

<a href="#"><img src="img/sales/2.jpg" alt="" /></a>

<div class="add-actions text-center">

<div class="project-dec">

<a class="venobox" data-gall="myGallery" href="img/sales/2.jpg">

<h4>Handbags and clutches</h4>

<span>30% OFF</span>

</a>

</div>

</div>

</div>

</div>

</div>

<!-- single-awesome-project end -->

<!-- single-awesome-project start -->

<div class="col-md-4 col-sm-4 col-xs-12 photo development">

<div class="single-awesome-project">

<div class="awesome-img">

<a href="#"><img src="img/sales/3.jpg" alt="" /></a>

<div class="add-actions text-center">

<div class="project-dec">

<a class="venobox" data-gall="myGallery" href="img/sales/3.jpg">

<h4>Redmi on sales</h4>

</a>

</div>

</div>

</div>

</div>

</div>

<!-- single-awesome-project end -->

</div>

</div>

</div>

</div>

<!-- awesome-portfolio end -->

<!-- our-skill-area start -->

<div class="our-skill-area fix hidden-sm">

<div class="test-overly"></div>

<div class="skill-bg area-padding-2">

<div class="container">

<!-- section-heading end -->

<div class="row">

<div class="skill-text">

<!-- single-skill start -->

<div class="col-xs-12 col-sm-3 col-md-3 text-center">

<div class="single-skill">

<div class="progress-circular">

<input type="text" class="knob" value="0" data-rel="85" data-linecap="round" data-width="175" data-bgcolor="#fff" data-fgcolor="#3EC1D5" data-thickness=".20" data-readonly="true" disabled>

<h3 class="progress-h4">New Year sales</h3>

</div>

</div>

</div>

<!-- single-skill end -->

<!-- single-skill start -->

<div class="col-xs-12 col-sm-3 col-md-3 text-center">

<div class="single-skill">

<div class="progress-circular">

<input type="text" class="knob" value="0" data-rel="80" data-linecap="round" data-width="175" data-bgcolor="#fff" data-fgcolor="#3EC1D5" data-thickness=".20" data-readonly="true" disabled>

<h3 class="progress-h4">Xmas sales</h3>

</div>

</div>

</div>

<!-- single-skill end -->

<!-- single-skill start -->

<div class="col-xs-12 col-sm-3 col-md-3 text-center">

<div class="single-skill">

<div class="progress-circular">

<input type="text" class="knob" value="0" data-rel="70" data-linecap="round" data-width="175" data-bgcolor="#fff" data-fgcolor="#3EC1D5" data-thickness=".20" data-readonly="true" disabled>

<h3 class="progress-h4">Diwali Sales</h3>

</div>

</div>

</div>

<!-- single-skill end -->

<!-- single-skill start -->

<div class="col-xs-12 col-sm-3 col-md-3 text-center">

<div class="single-skill">

<div class="progress-circular">

<input type="text" class="knob" value="0" data-rel="60" data-linecap="round" data-width="175" data-bgcolor="#fff" data-fgcolor="#3EC1D5" data-thickness=".20" data-readonly="true" disabled>

<h3 class="progress-h4">Dushera Sales</h3>

</div>

</div>

</div>

<!-- single-skill end -->

</div>

</div>

</div>

</div>

</div>

<!-- our-skill-area end -->

<!-- Start Footer bottom Area -->

<footer>

<div class="footer-area">

<div class="container">

<div class="row">

<div class="col-md-4 col-sm-4 col-xs-12">

<div class="footer-content">

<div class="footer-head">

<div class="footer-logo">

<h2>Flipkart</h2>

</div>

<p>Commercial sales and offers where you can meet all the accessories of goods and products</p>

<div class="footer-icons">

<ul>

<li>

<a href="#"><i class="fa fa-facebook"></i></a>

</li>

<li>

<a href="#"><i class="fa fa-twitter"></i></a>

</li>

<li>

<a href="#"><i class="fa fa-google"></i></a>

</li>

</ul>

</div>

</div>

</div>

</div>

<!-- end single footer -->

<div class="col-md-4 col-sm-4 col-xs-12">

<div class="footer-content">

<div class="footer-head">

<h4>information</h4>

<div class="footer-contacts">

<p><span>Tel:</span> +123 456 789</p>

<p><span>Email:</span> srisimhadri345@gmail.com</p>

<p><span>Working Hours:</span> 9am-5pm</p>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</footer>

<a href="#" class="back-to-top"><i class="fa fa-chevron-up"></i></a>

<!-- JavaScript Libraries -->

<script src="lib/jquery/jquery.min.js"></script>

<script src="lib/bootstrap/js/bootstrap.min.js"></script>

<script src="lib/owlcarousel/owl.carousel.min.js"></script>

<script src="lib/venobox/venobox.min.js"></script>

<script src="lib/knob/jquery.knob.js"></script>

<script src="lib/wow/wow.min.js"></script>

<script src="lib/parallax/parallax.js"></script>

<script src="lib/easing/easing.min.js"></script>

<script src="lib/nivo-slider/js/jquery.nivo.slider.js" type="text/javascript"></script>

<script src="lib/appear/jquery.appear.js"></script>

<script src="lib/isotope/isotope.pkgd.min.js"></script>

<!-- Contact Form JavaScript File -->

<script src="contactform/contactform.js"></script>

<script src="js/main.js"></script>

</body>

</html>

**5.2 IOT page code**

**Html code for IOT web page**

<html>

<head>

<title>

Atmosphere Data

</title>

<style>

body {

text-align:center;

}

.sensor{

background-color: black;

padding: 20px;

font-size: 35px;

color: white;

margin: 10px;

padding: 10px;

}

table, th, td {

border: 2px solid black;

border-collapse: collapse;

padding: 20px;

margin: 20px;

margin-left:auto;

margin-right:auto;

}

</style>

</head>

<body onload="updateValues()">

<div class="sensor">

<h2 style=" text-align: center;">Atmosphere</h2>

</div>

<table style="width:50%" bgcolor="gainsboro" align="center">

<tr>

<th><h3 id="temperature">Temperature : </h3></th>

</tr>

<tr>

<th><h3 id="humidity">Humidity : </h3></th>

</tr>

</table>

<script>

function updateValues() {

setInterval(function() {

getData();

},4000);

}

function getData() {

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange=function() {

if (this.readyState == 4 && this.status == 200) {

let response=JSON.parse(this.response);

document.getElementById("temperature").innerHTML = "Temperature : " + response.temperature;

document.getElementById("humidity").innerHTML = "Humidity : " + response.humidity;

}

};

xhttp.open("GET", "/api/getdata", true);

xhttp.send();

}

function putData(temp,humd){

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange=function() {

};

xhttp.open("POST", "/api/putdata", true);

xhttp.setRequestHeader("Content-type", "application/json");

xhttp.send(JSON.stringify({temperature:temp,humidity:humd}));

}

</script>

</body>

</html>