Akhila Nair

Cloud Computing – Sec 031

Exercise #2

INDIAN TIFFIN SERVICE:

Exercise two in Cloud Computing class gave me an opportunity to work with Amazon AWS. I had always heard about it and never had tried it myself. I started with what is AWS. Amazon web services is a platform where you can create, develop and put in use different cloud computing services according to specific need and demands. I started with reading different papers of previous students. This helped me understand the registering process, though it had some minor changes this year, it was easy to achieve. With the UMass Lowell code you get 100$ credit to use the services.

If you are new to the Cloud Computing I would suggest, start with reading the book: Amazon Web Services in a Month of Lunches. It explains the very basic step of understanding how to create an instance. I played with creating an instance or two and checked uses of different Operating systems that can be used.

Project Objective:

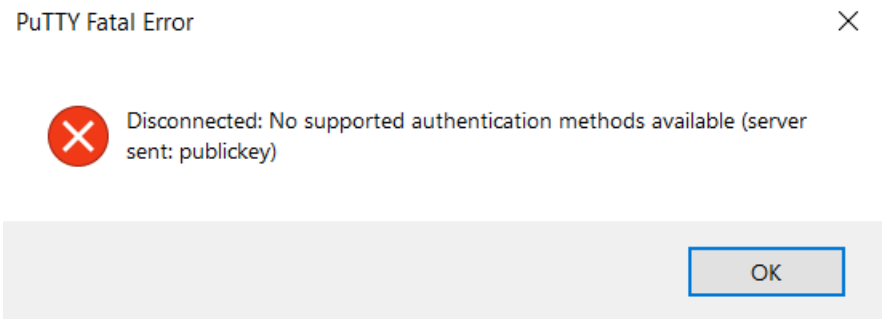
The project aims at using different services, with end product being a functional website. Being an international student and missing home food, I grabbed this opportunity to make a Website where we can get Indian home cooked Food. The product Site will be called Indian Tiffin Services. The list of Amazon web Services that I have used for this project are:

1) EC2 – Tier 1
2) Elastic Ip – Tier 2
3) S3 – Tier 1
4) Cloud Formation – Tier 1
5) IAM – Tier 1
6) CloudWatch – Tier 2
7) Billing Alarm – Tier 1
8) SNS – Tier 2
9) Trusted Advisor – Tier 1 (New)

Prerequisite:

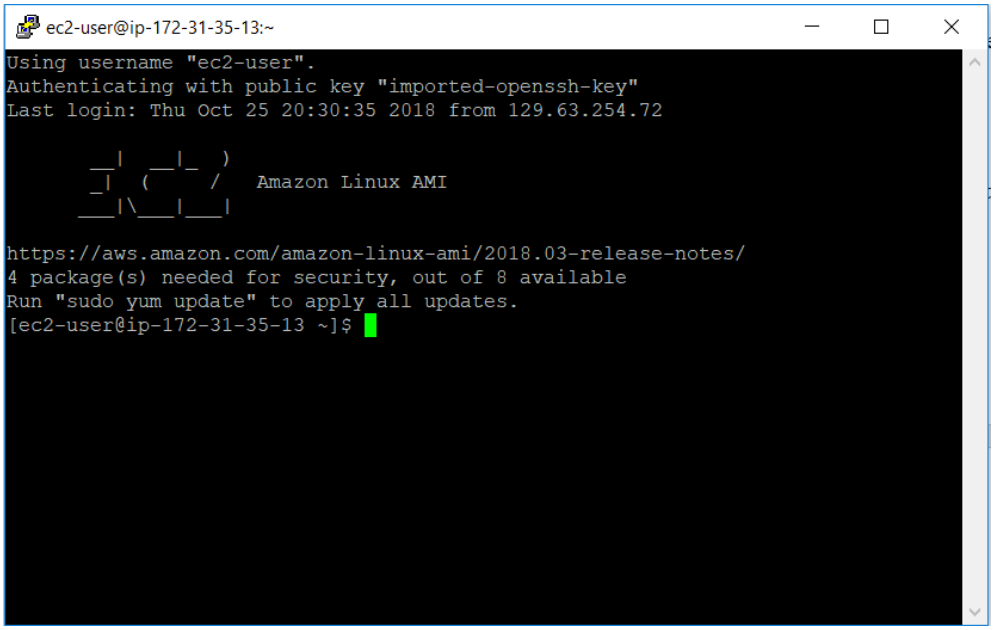
If you are a windows user, I would recommend downloading the latest version of Putty. This will help one access the instance remotely.

This is an important step and I was stuck in this step for a considerable amount of time. I got an error called “Putty Fatal Error”. This meant that my authentication was not correct and hence couldn’t remotely SSH to my instance. So, I am putting down the steps before hand for future reference.



The steps to access an ec2 instance are:

1) Save the Key pair (.pem) file in your home directory.
2) To convert it to .ppk file open the Putty Generator (and not Putty Terminal).
3) Click on “Load” and choose the parameter as RSA.
4) Select all files and load your key pair.
5) After converting it to .ppk file, open the Putty terminal.
6) Put the Hostname as ec2-user@publicIp_Addr_of_Your_Instance (For example: ec2-user@18.225.1.54)
7) On the left Navigation Panel click on “SSH” and double Click on “Auth”.
8) Browse the Private Key file you converted in step 5 and click on “Open”.
9) The first time you log in to an instance it will ask if you trust this address, click on “yes” and continue.
10) This should open a Putty terminal window as shown below.



IAM Role:

One of the requirements of the project was to create an Administrator access for Prof Bob.

To go about it, Select IAM services form the list of services of AWS. Click on “Add User”. Give the User name and define the access you would like to give and choose custom password. Next click on “Permissions” to grant access to the user. Here Choose “Set Permissions boundary” and then select – Use a permissions boundary to control the maximum user permissions. From the List of permissions, choose administrator access for this user. This will provide Prof. Bob full access of your AWS account and resources. Click on “Review” and then “Finish”.

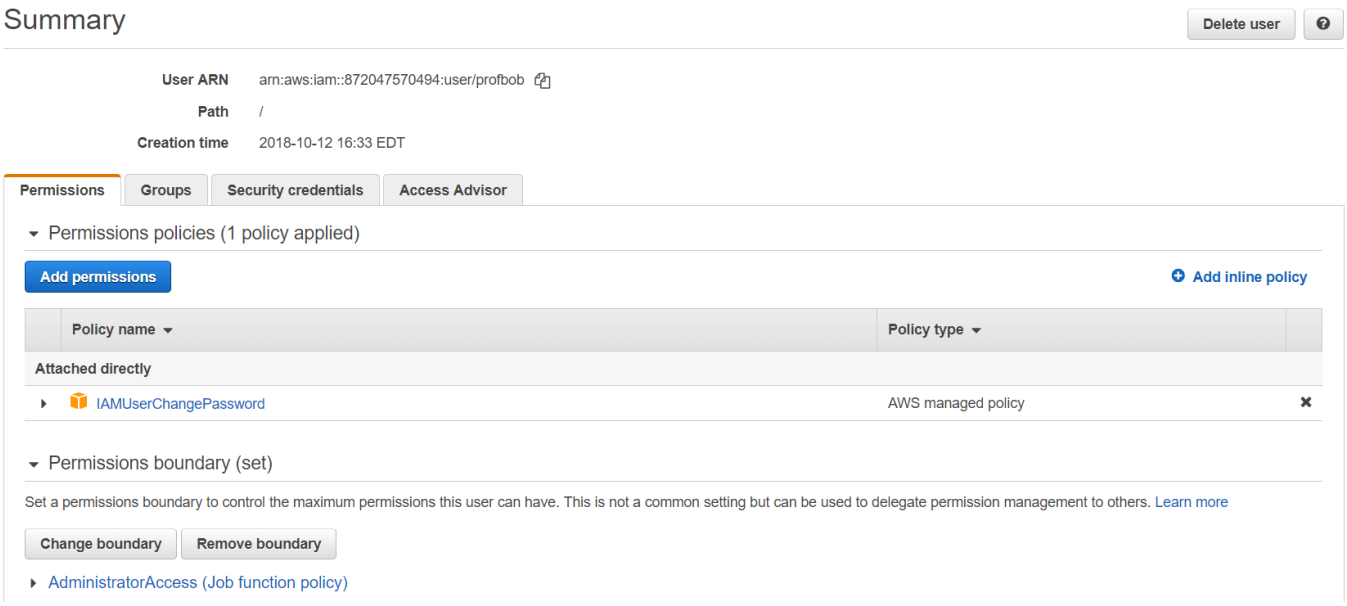
This should create a user ID and Password for the user.

User ID: profbob

Password: myproject1

The URL to sign in is:

https://872047570494.signin.aws.amazon.com/console



CloudFormation:

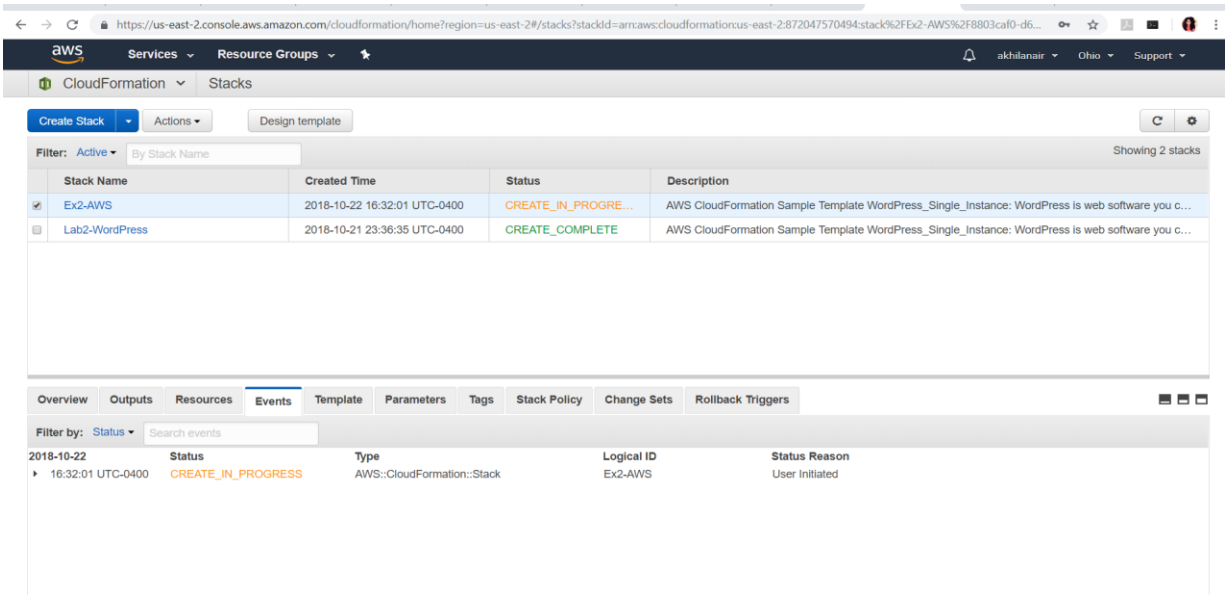
Version Control and keeping track of changes can be challenging. Things get even harder if you must develop your infrastructure stack multiple times for testing purposes. CloudFormation creates and manages the infrastructure stack and application stack in a control and predictable way. It provisions and manages information stacks of AWS resources of templates the user wants to create and modify. It is easy to manage anything, be it a single EC2 instance or a multi-level application. It creates a blue-print of the template.

The reason I tried CloudFormation is that by default creates an ec2 instance with WordPress blog as template.

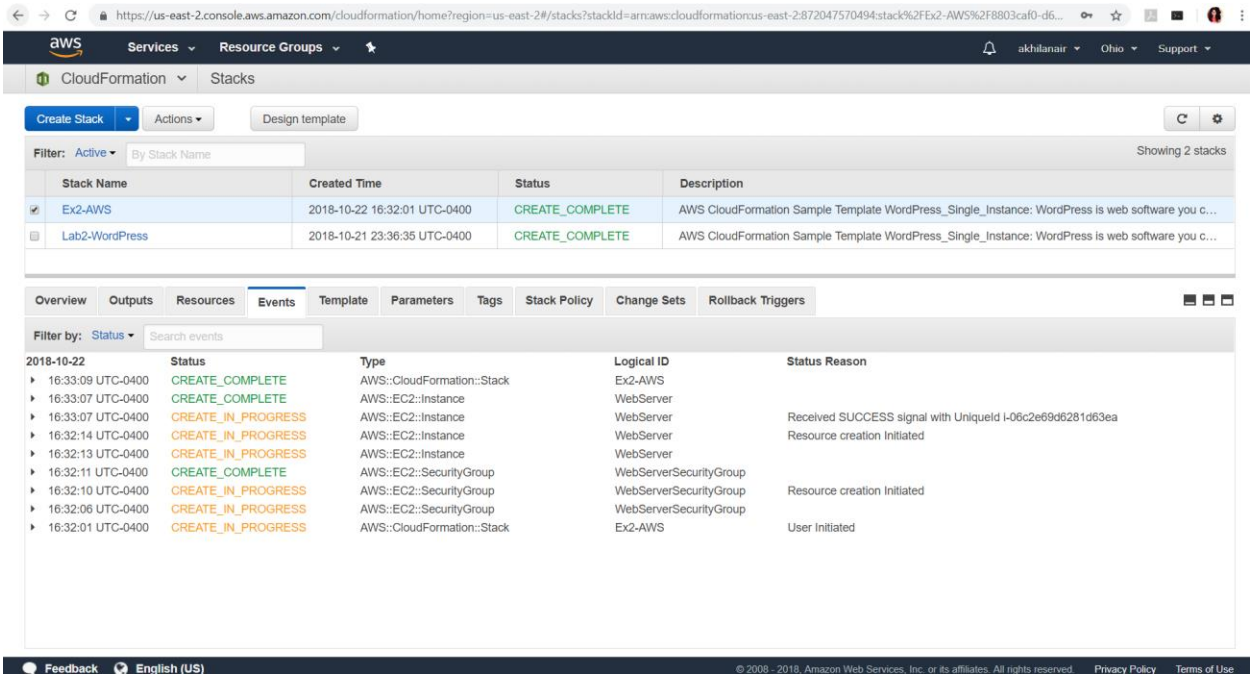
For creating one:

- Search for CloudFormation Service in the AWS dashboard.
- Click on Create Stack and choose "select a sample template".
- On the Next page:
  Stack Name: EX2-AWS
  DB Name: wordpressdb
  DB Password: AkhilaNair14 (choose this carefully as they have set conventions for Password)
  DB Root Password: AkhilaNair14 (choose this carefully as they have set conventions for Password)
  DB User: anair
  Instance type: t2.micro (Be careful to select this, as yje default selected is not free tier eligible).
  Key Name: myawsproject
- In the options page, keep the default values as it is.
- Click on review, check all parameters and create the stack.

At the start it will look something like this:



After a minute or two the page should change to this:



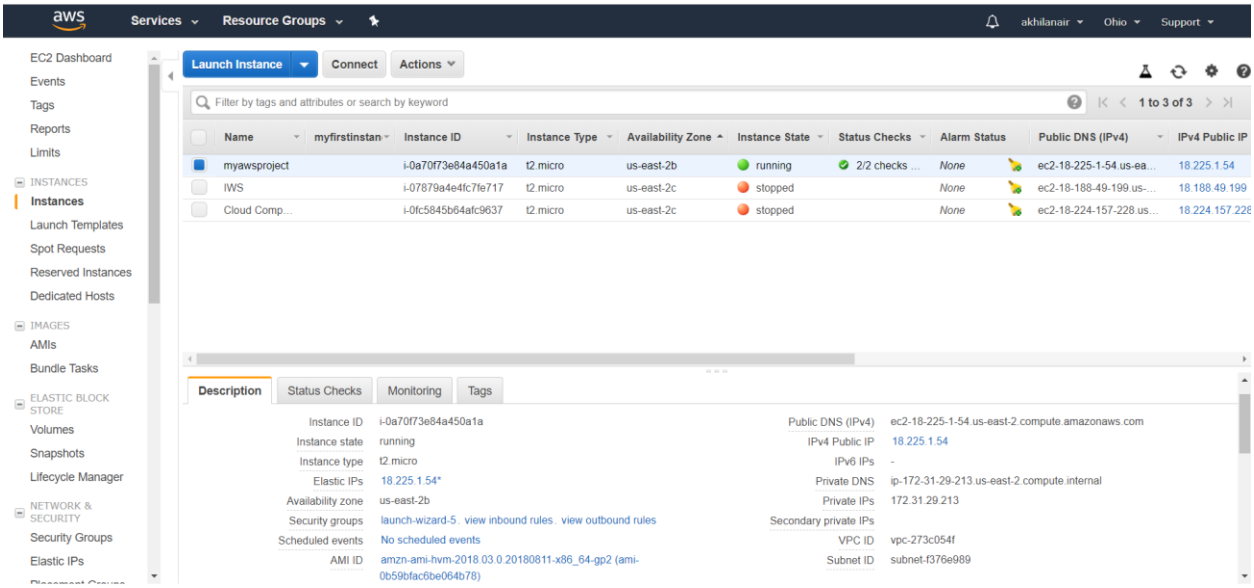
Now when you look into instances tab, you will be able to see an Amazon Linux Instance.

Elastic IP:

Each time you log in to your Ec2 instance, the public Ip address of your instance is changed. This also means, each time you must SSH with a different IP address. Considering that my website should have a static IP to logon I considered changing the Public Ip to an elastic IP.

This can be achieved in 5 easy steps:

- In the navigation panel search for Elastic IP.
- Click on “Allocate New Address” and then select “Amazon pool”.
- Finally Click on allocate address. A new Elastic Ip will then be created.
- Select that address and click on “Action” and then “Associate address”.
- Finally select the instance you want to allocate the new address to.



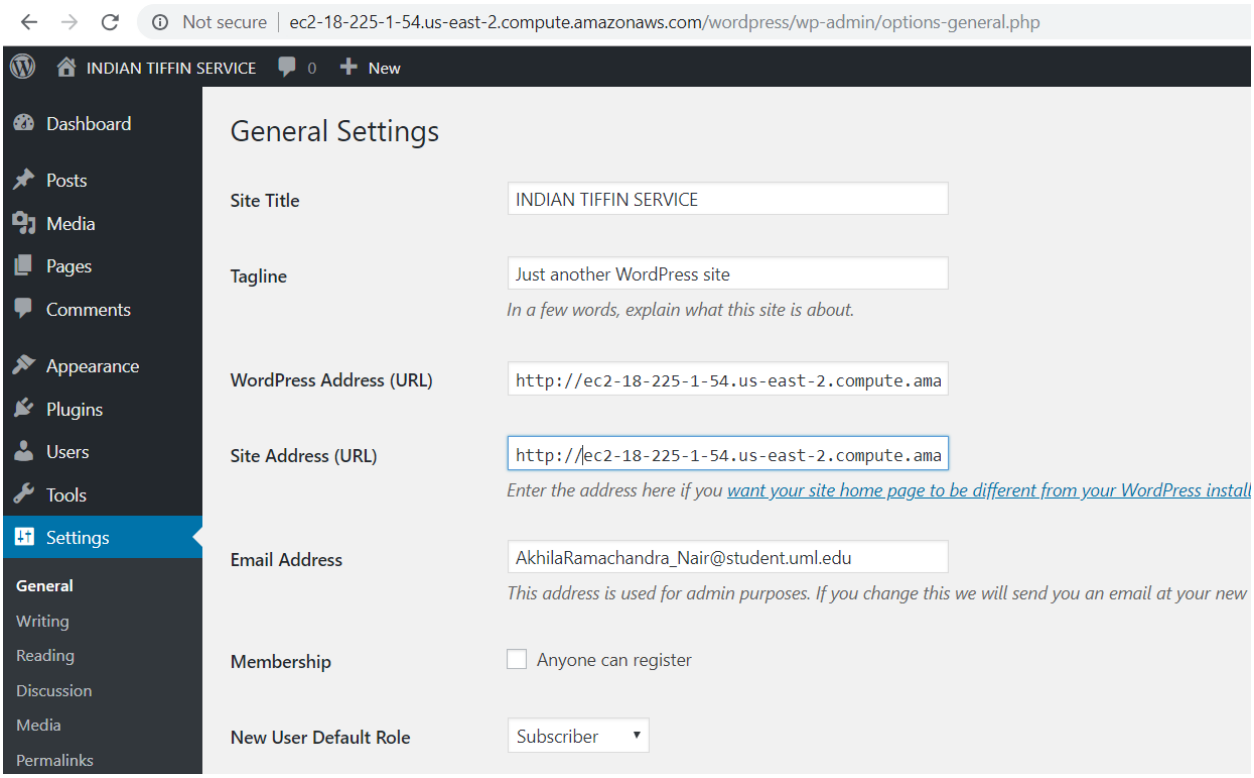
Now when we follow the link of public DNS it should take to the WordPress installation page. The link for my project is:

ec2-18-225-1-54.us-east-2.compute.amazonaws.com

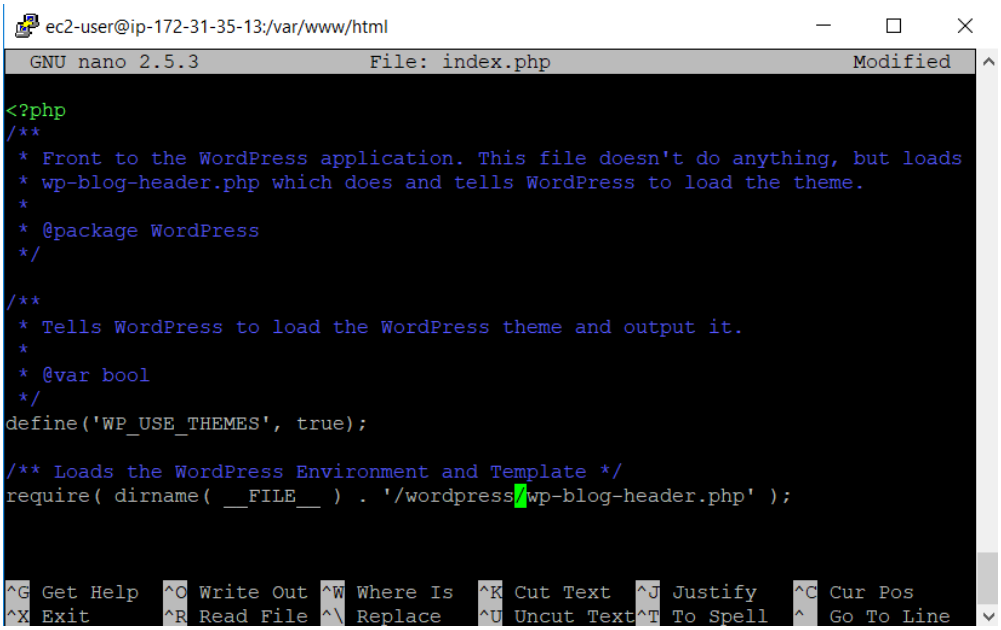
The problem I faced here was that, with this link it took me to the Apache test page and not the WordPress installation page. If I added /wordpress/ along with this link then it directed me to WordPress installation page. After searching and reading number of papers I realized that I had to copy the index.php file to root directory of var/www/html/

The steps to achieve this are:

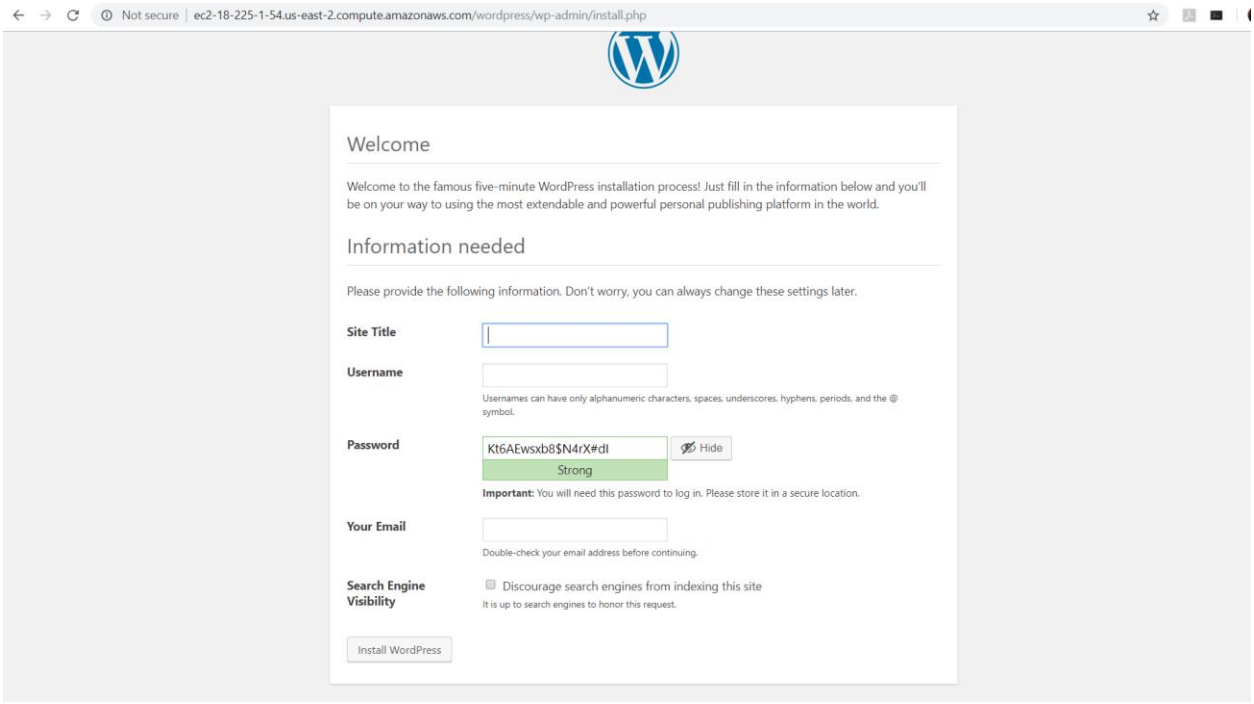
- Go to your WordPress Dashboard -> General setting.
- Copy the DNS IP address to Site Address URL and add the http:// in the beginning.



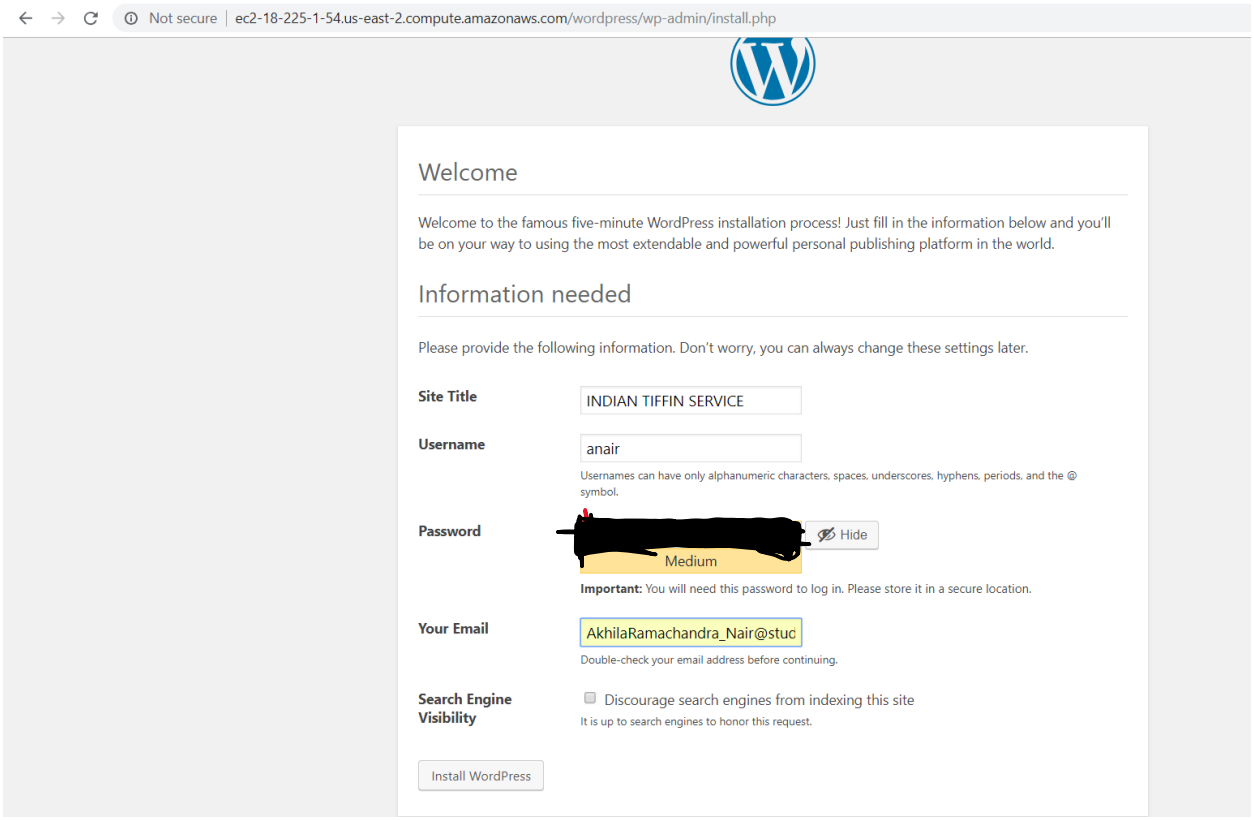
- Save the changes.
- Go to var/www/html/wordpress/ folder.
- Copy the index.php file and paste it in var/www/html/ folder.
- Open it with the nano editor and change the following
  require(dirname(__FILE__) . '/wp-blog-header.php'); to
  require(dirname(__FILE__) . '/wordpress/wp-blog-header.php');



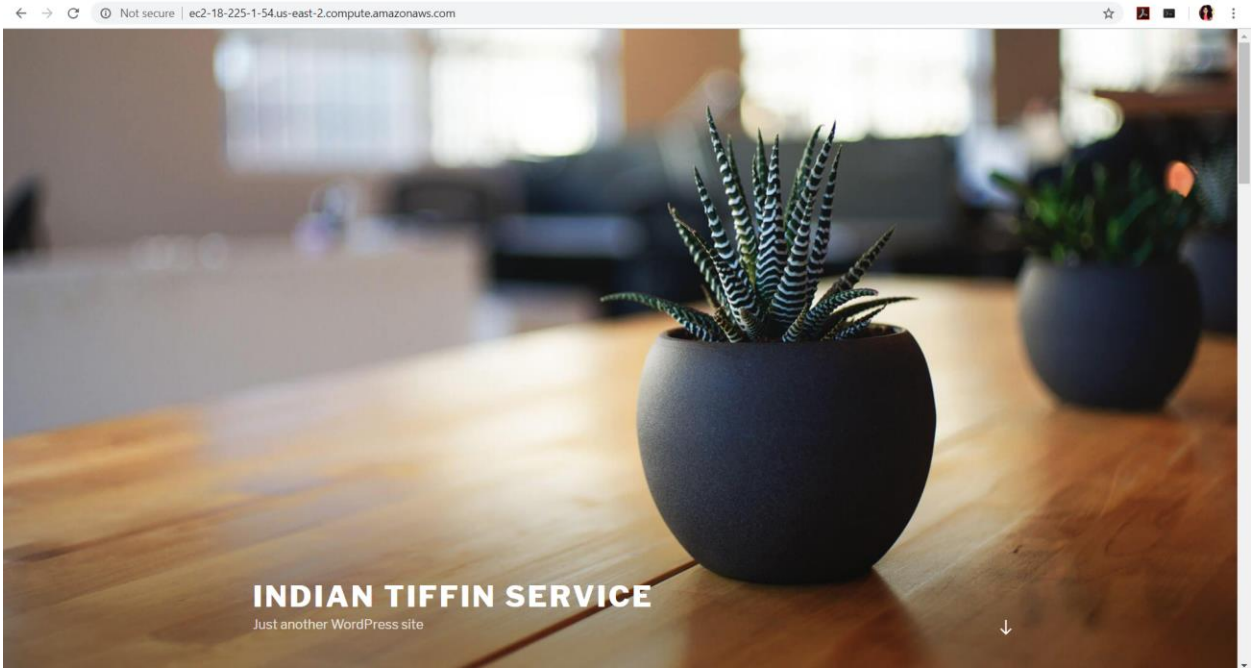
- Save the file and exit.
- Now when you copy the DNS IP as the URL you will now reach the WordPress Installation Page.



Put in the desired information and select “Install WordPress”.



Finally, when you login with your ID and password you will be able to see your website.



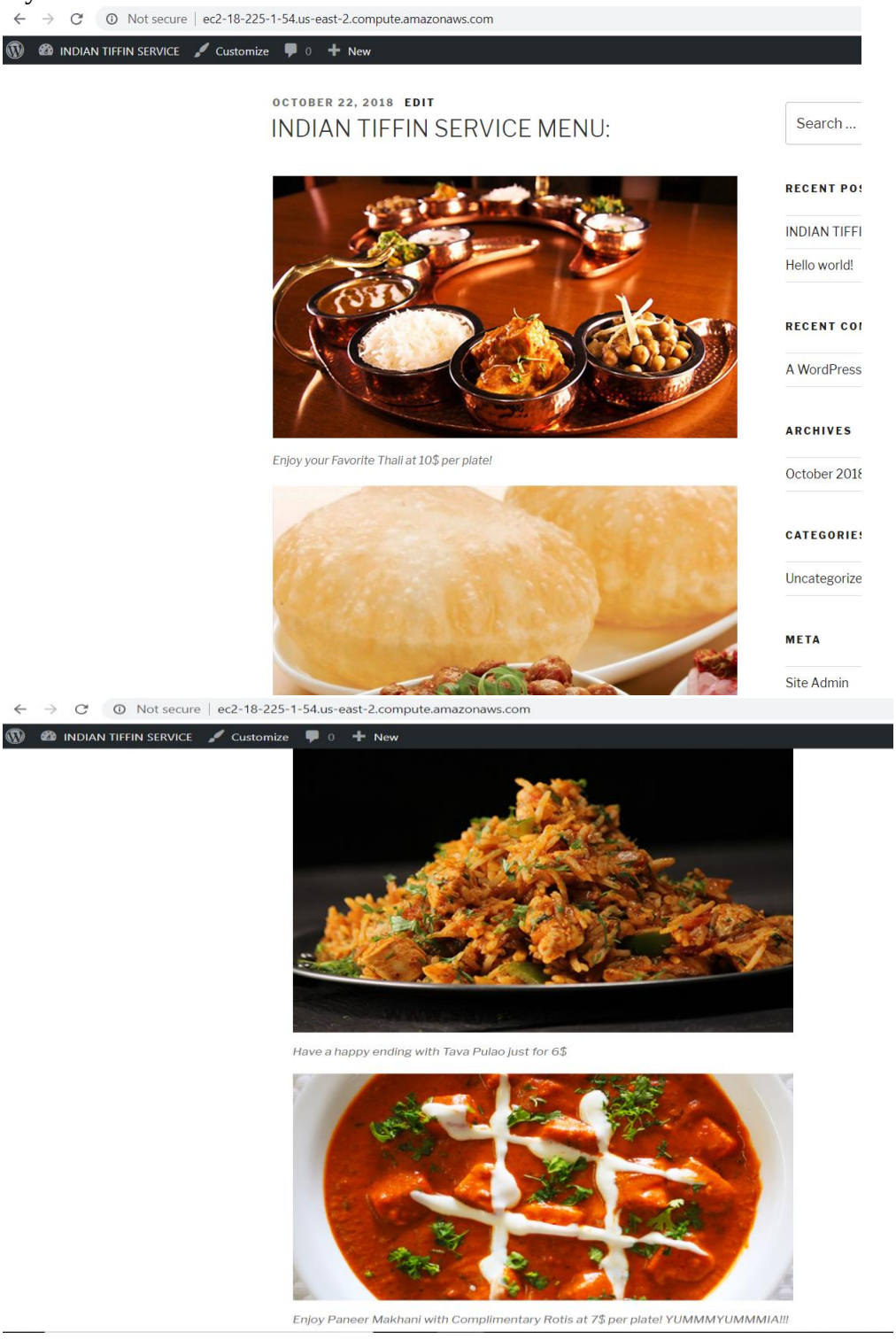
S3:

As specified by the name, Simple Storage services are used for storing any of your documents or files. This makes computing easier for developers. The data stored in S3 bucket can be retrieved from anywhere and at any given time. This increases the efficiency and accessibility. The storage space of your application also can be used for some other purpose because, you can link the files or images to the S3 bucket. Security measurements can also ne added if they want a secure access to the bucket.

The process of creation of the S3 bucket is:

- From the list of Amazon web services, search for S3.
- Click on "Create Bucket" and enter the bucket name.
- Be careful when you choose the Region, it should be in the same region where your instances that are going to access these buckets are deployed. (I created a bucket in different region for test purpose and used the same to link it to my WordPress Website. This didn't fetch my images from the bucket because the regions differed).
- Keep the default configuration parameters for this purpose.
- Set the permissions according to your system needs.
- Click on Review and create.
- You can now add some images or files in this bucket to be accessed later.
- I added pictures of some food that I had taken and saved it in the bucket.
- I then went to my WordPress dashboard and clicked on add a post.
- I entered the title as "Indian Tiffin Service Menu:"

- In the add media section, I selected “Add from URL” (My S3 bucket).
- You can add link test if you want and you can then hit “Post”.
- With this step, I successfully added a bunch of images to my website through the link to my S3 bucket.

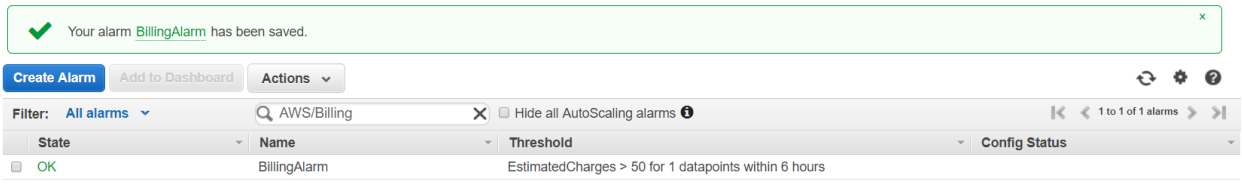


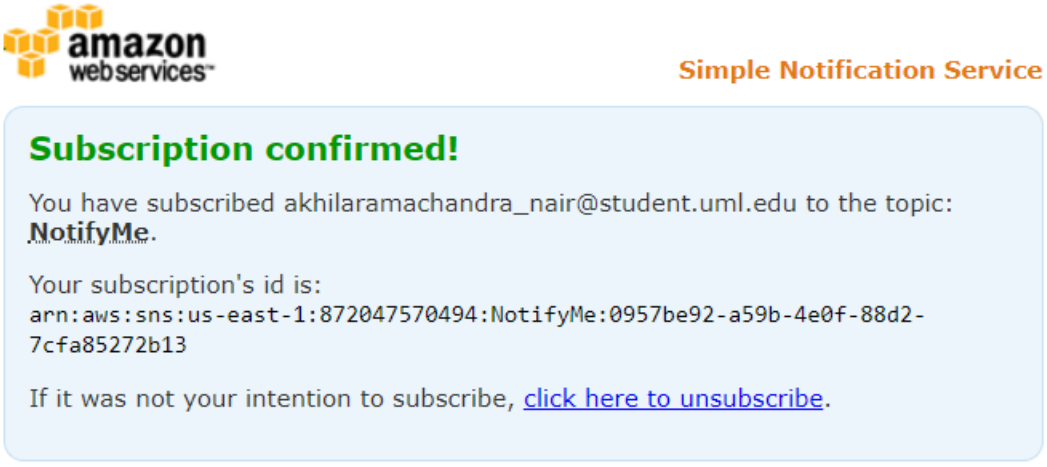
Creating a billing alarm:

I love this feature of Amazon, this is far by one of the best services to use. No one wants to be woken up by a surprise with a bill they didn't accept for. AWS allows you to create such billing alerts which can be formulated by yourself. The moment your specified component is above the threshold, it will send you an alert regarding the same in your specified email ID. There are chances that you forgot to stop your instance after the use and you realize it a week later? No problem, Amazon CloudWatch got your back!

For creating a billing alert:

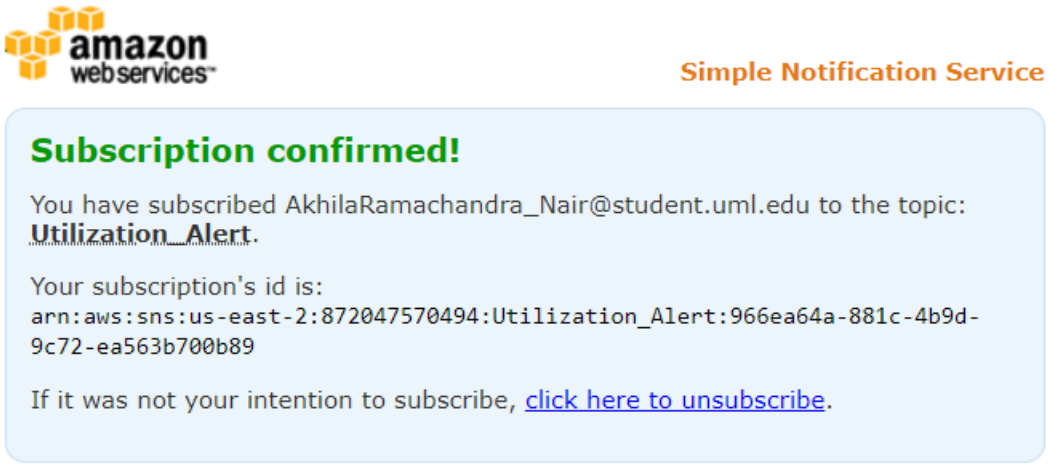
- You have to first activate billing alerts, which can be done by navigating through Account Name → Billing Dashboard →Activate alerts.
- After this step, search for CloudWatch service from the AWS services dashboard and click on Billing and follow the commands to create a billing alarm. It uses SNS service to send a notification to the registered email-id.

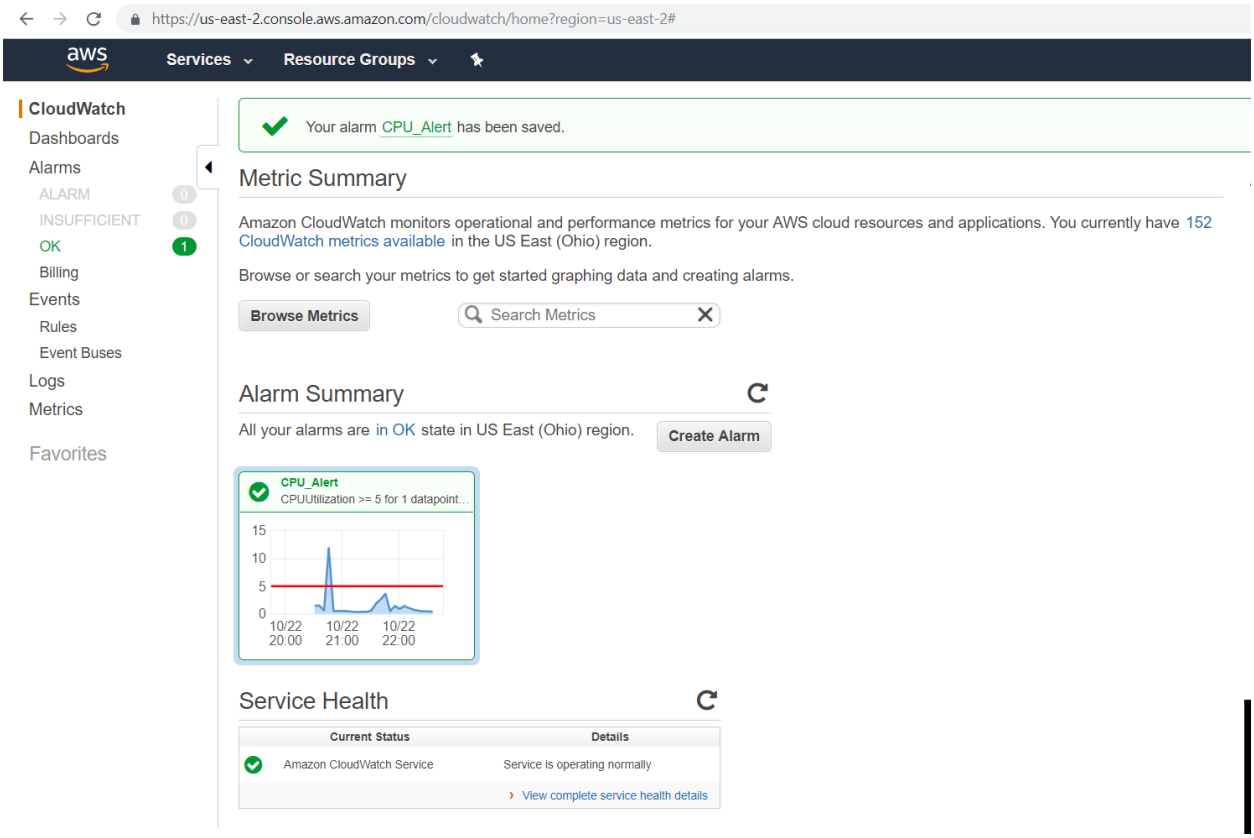




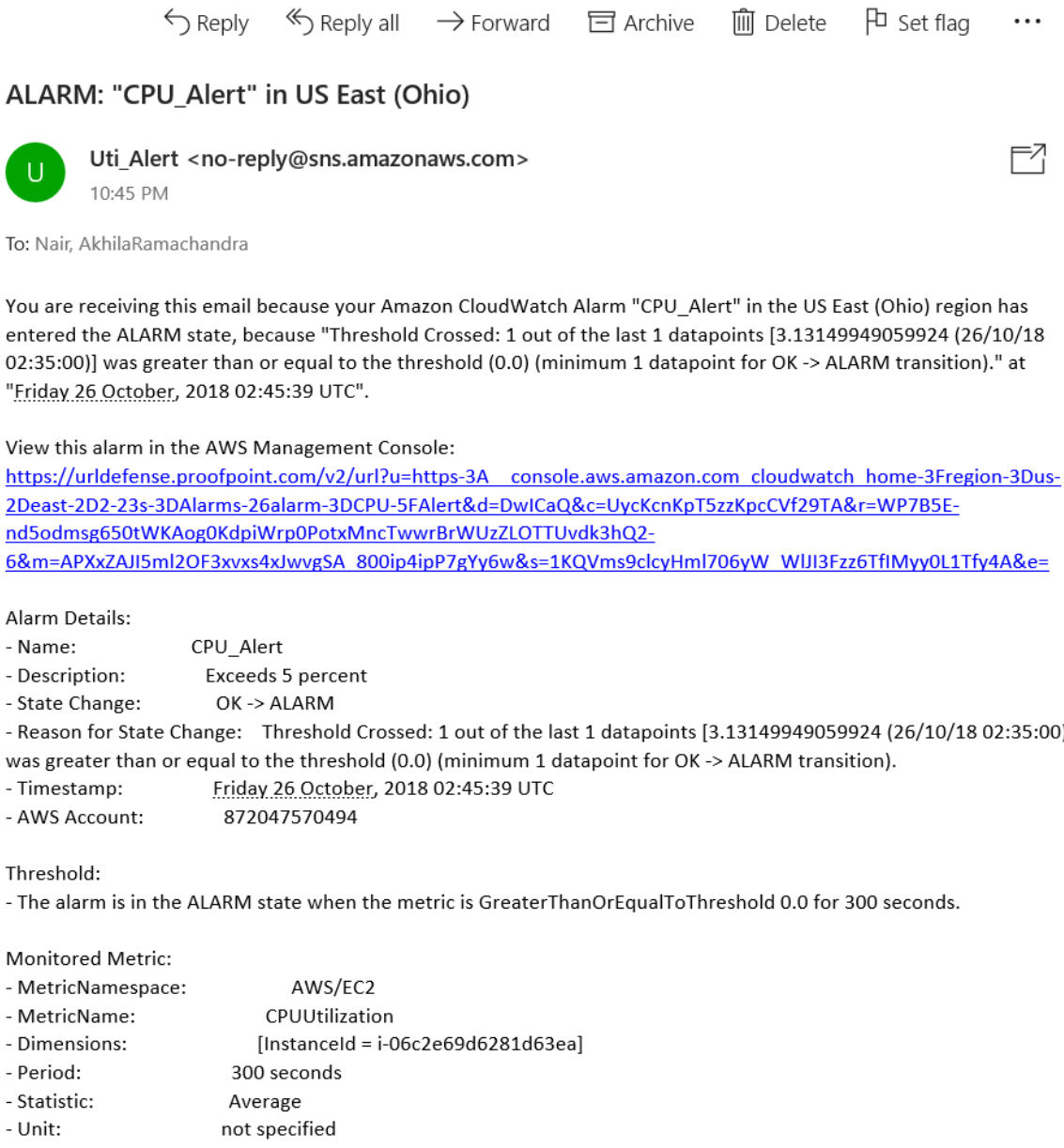
- If you want to create a CPU utilization alert, click on "Alarms" under the CloudWatch page.
- Click on "Create Alarms", Choose Ec2 metrics and select the instance you are working on.

- Input the threshold, and minimum value.
- Give a suitable name to the alert and click ok.
- You will get a confirmation dialogue box saying that subscription is confirmed.





I also got an alert when I changed the minimum threshold value for the demonstration purpose:



Reply Reply all Forward Archive Delete Set flag ...

ALARM: "CPU_Alert" in US East (Ohio)

U **Uti_Alert <no-reply@sns.amazonaws.com>**
10:45 PM

To: Nair, AkhilaRamachandra

You are receiving this email because your Amazon CloudWatch Alarm "CPU_Alert" in the US East (Ohio) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [3.13149949059924 (26/10/18 02:35:00)] was greater than or equal to the threshold (0.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Friday 26 October, 2018 02:45:39 UTC".

View this alarm in the AWS Management Console:
https://urldefense.proofpoint.com/v2/url?u=https-3A__console.aws.amazon.com_cloudwatch_home-3Fregion-3Dus-2Deast-2D2-23s-3DAlarms-26alarm-3DCPU-5FAlert&d=DwICaQ&c=UycKcnKpT5zzKpcCVf29TA&r=WP7B5E-nd5odmsg650tWKAog0KdpiWrp0PotxMncTwwrBrWUzZLOTTUvdk3hQ2-6&m=APXxZAJI5ml2OF3xvxs4xJwvgSA_800ip4ipP7gYy6w&s=1KQVms9clcyHml706yW_WlJI3Fzz6TfIMyy0L1Tfy4A&e=

Alarm Details:
- Name: CPU_Alert
- Description: Exceeds 5 percent
- State Change: OK -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [3.13149949059924 (26/10/18 02:35:00)] was greater than or equal to the threshold (0.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Friday 26 October, 2018 02:45:39 UTC
- AWS Account: 872047570494

Threshold:
- The alarm is in the ALARM state when the metric is GreaterThanOrEqualToThreshold 0.0 for 300 seconds.

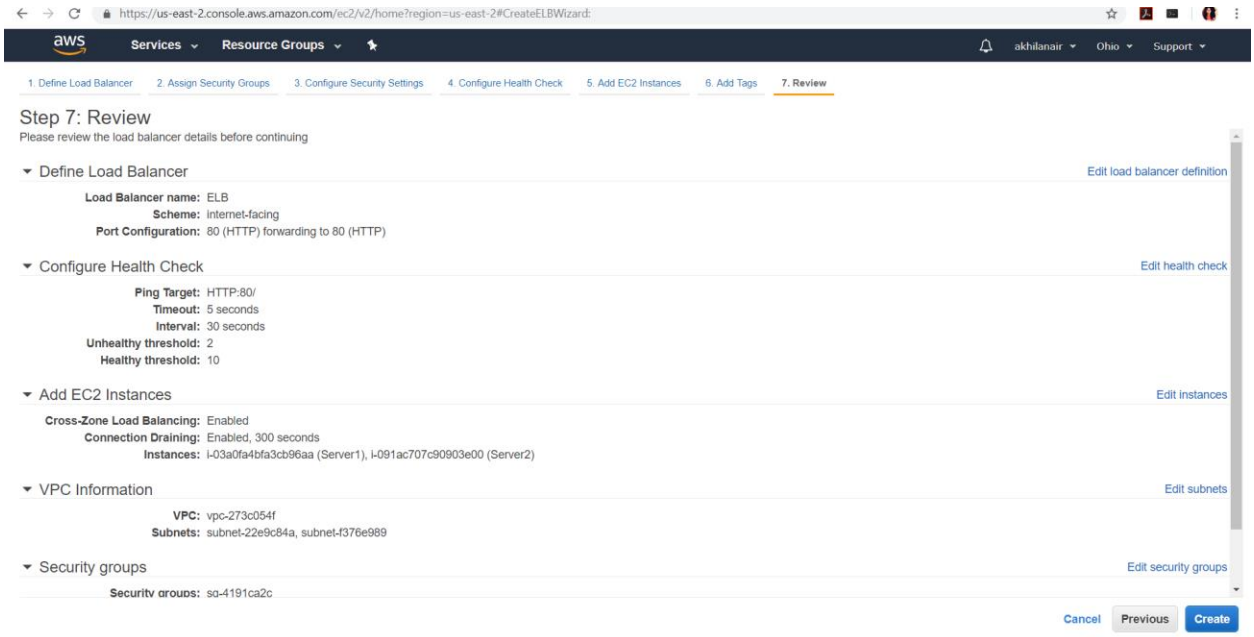
Monitored Metric:
- MetricNamespace: AWS/EC2
- MetricName: CPUUtilization
- Dimensions: [InstanceId = i-06c2e69d6281d63ea]
- Period: 300 seconds
- Statistic: Average
- Unit: not specified

Elastic Load Balancer:

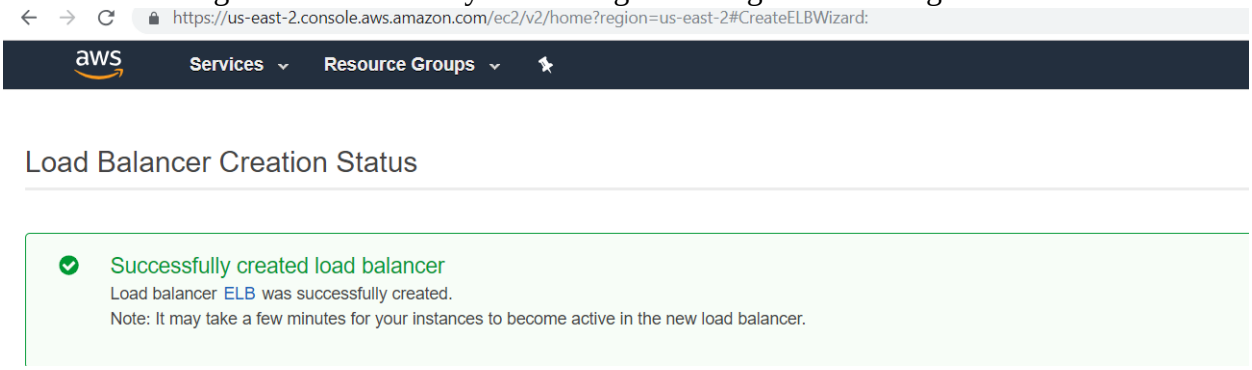
Load balancer will reduce the traffic at your address and send it to another site which replicates your first instance. It distributes the data network to avoid any cluster in just one particular port or instance. This helps in reducing the response time and thereby increase the efficiency of the network. A load balancer needs two EC2 instances with same security policies.

The steps for creating a load balancer are:

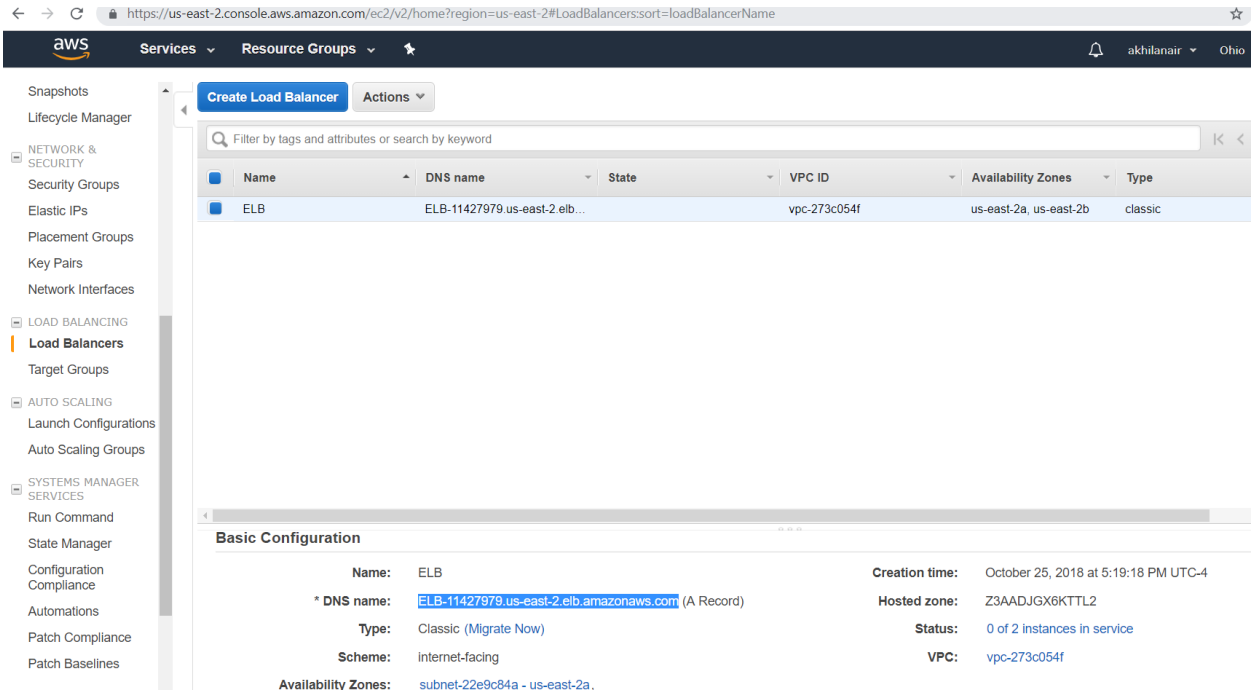
1) Create two instances with different names but same security parameters defined.
2) Then search for Load Balancer in the left navigation bar of AWS services and click on "Create Elastic Load Balancer".
3) There are three types of balancer, namely Application load balancer, Network Load balancer and Classic Load balancer.
4) Application load balancer is used when you have an application and need flexible features to set it up. It can configure routing and application parameters.
5) Network Load balancer is used when someone needs high performance or millions of requests per second.
6) The once we use is Classic load balancer. This one allows HTTP, HTTPS and TCP Connections.
7) Give the load balancer an appropriate name and then click on enable VPC configuration. Then choose at least two subnets from the options given below.
8) Select the security group as required. You can create a new security group or choose from existing security group.
9) You can skip the step of configuring security setting for this part and click on Health checks.
10) Keep the ping path as / so that it directs to index.html page.
11) Keep all the other configuration as it is.
12) Next step is to add the EC2 instance. Select the two instances and give tags if you need to.
13) Click on "review and create".



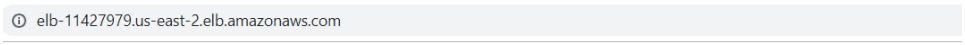
After creating the load balancer you should get a dialogue box stating:

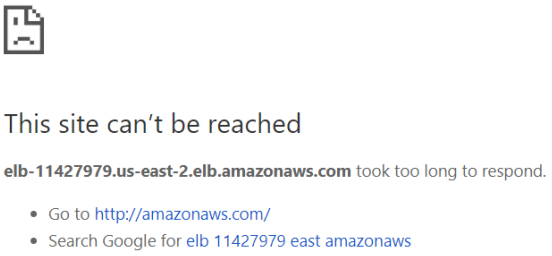


It may take a few minutes to change the status of the instance from Out of Service to active. After its active, you can copy the DNS URL of the elastic load balancer you just created and paste it in the URL section. The load balancer should now work.



It didn't work for me, I tried it n number of times. I am not able to understand what the error is, but whenever I try to access the DNS of Elastic Load balancer, I get the following message:

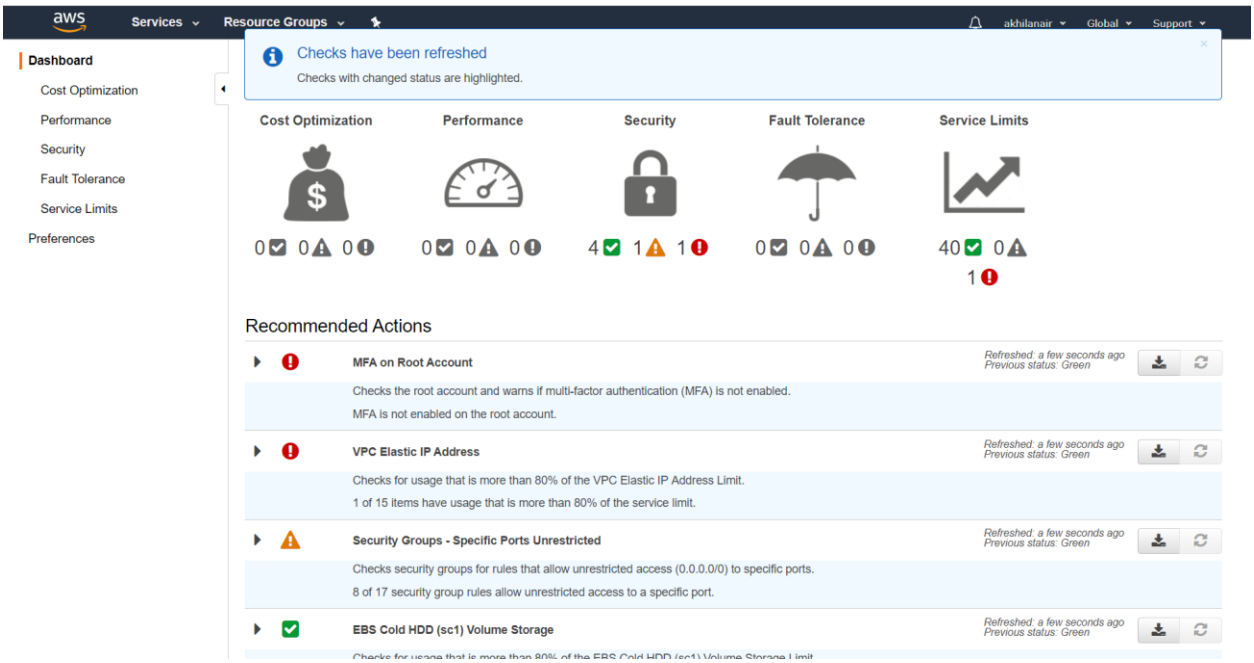




Trusted Advisor:

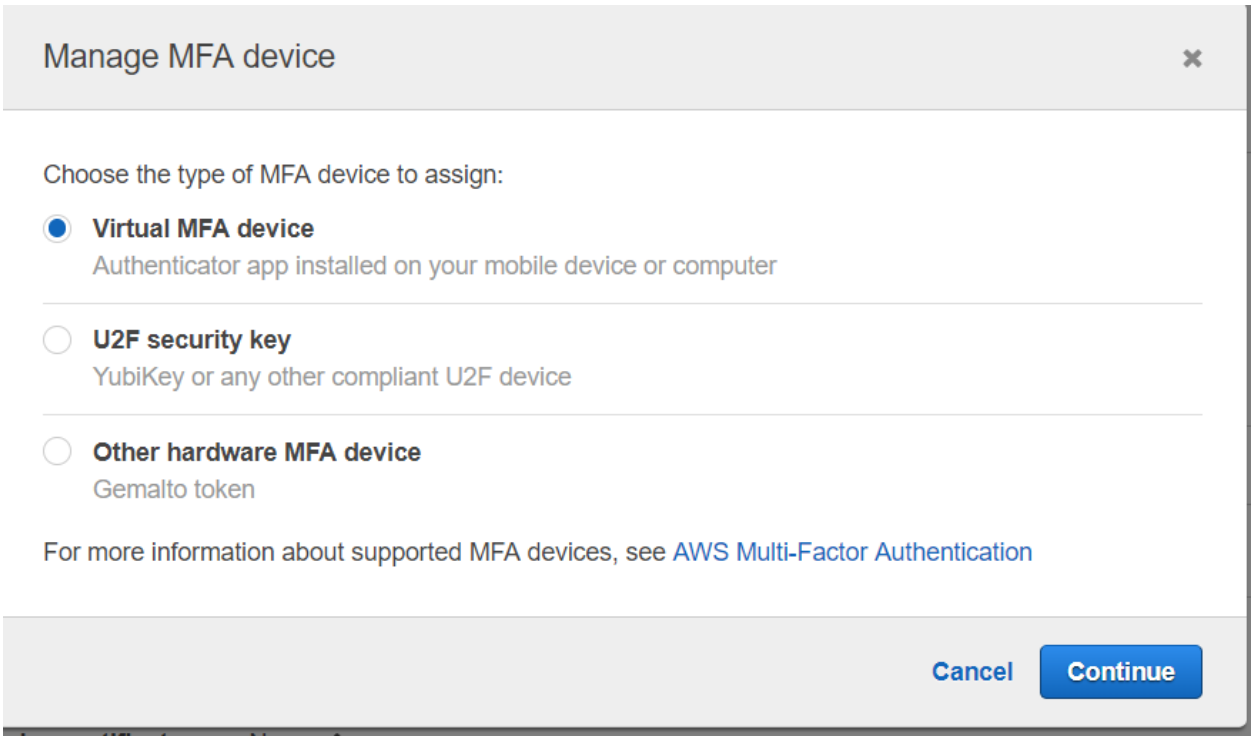
This feature of AWS helps you optimize your cost, performance, security, fault tolerance and service limits.

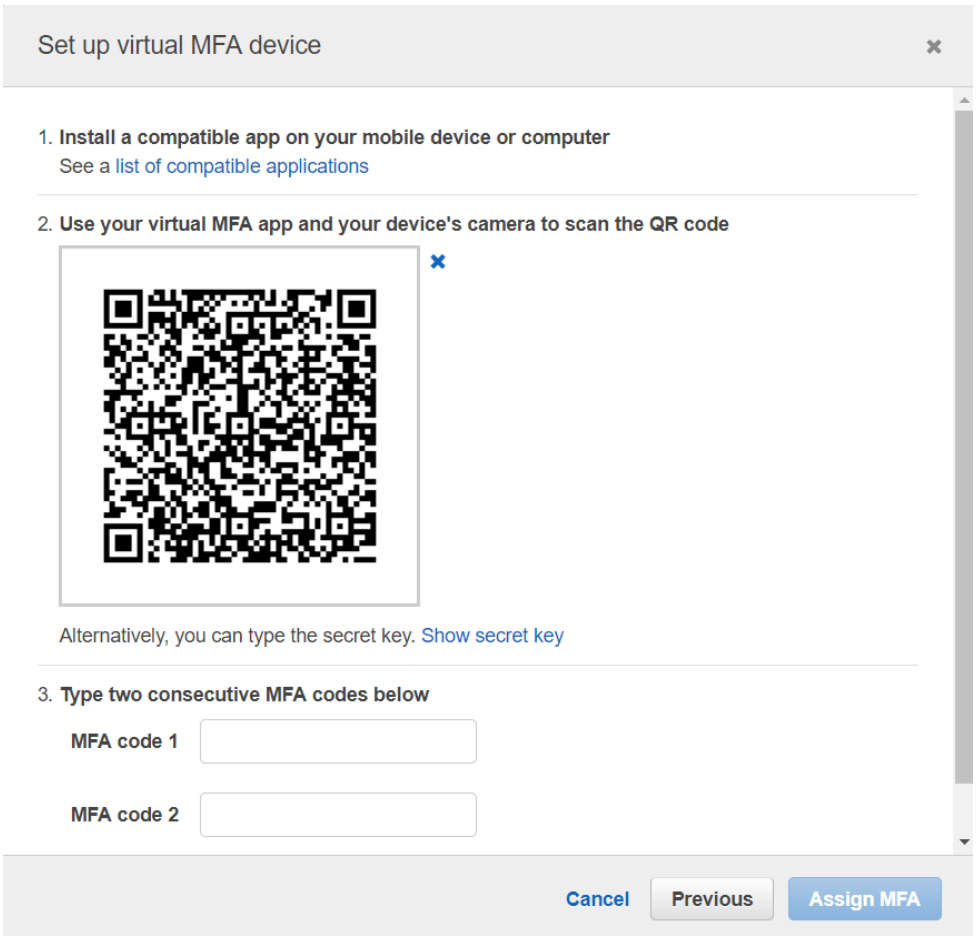
To create a trusted advisor feature, search for it in the dashboard. Without configuring it looks like this:



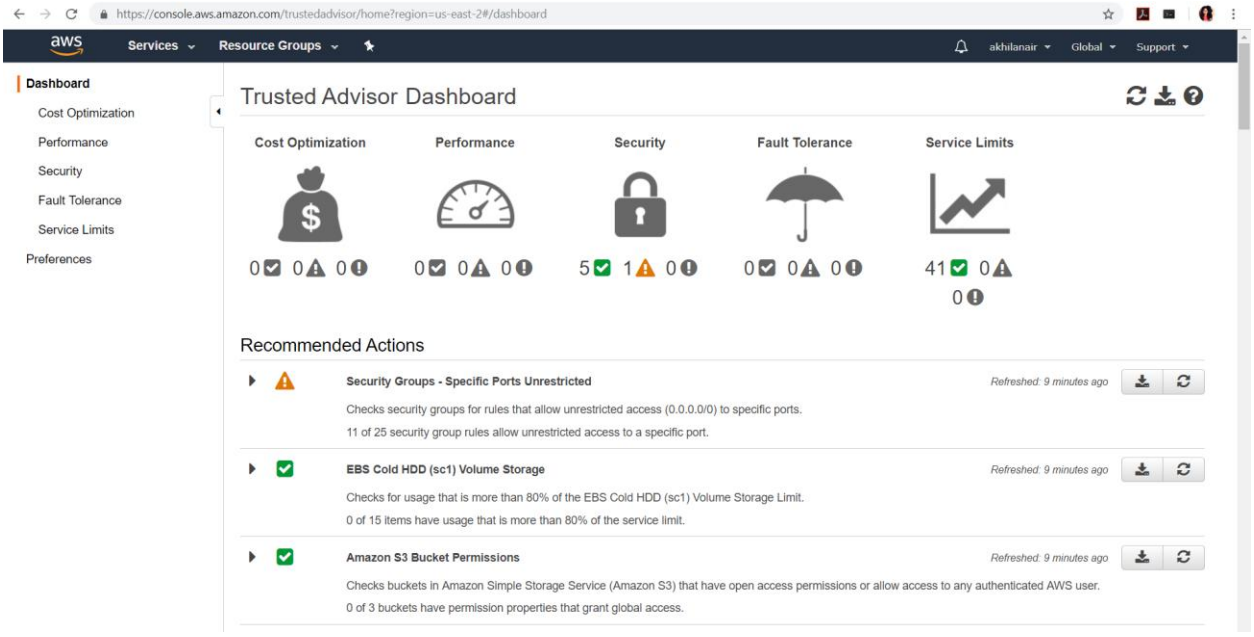
I realized I had consumed all the available IP addresses (maximum is 5). I released addresses which are not in use. This got rid of the VPC Elastic IP address recommended action.

I then created Virtual MFA on my root account by logging in to IAM and creating one. I downloaded the google authenticator app on my android phone. This feature gives a double authentication for logging in to the account keeping it safe from outside world and threats. It asks for barcode scanning and then we have to input two set of unique codes, after which it is authorized.





So, every time you login to the AWS account you have to provide MFA code.



After all the configuration trusted advisor looks like this.

Conclusion:

With this project, I could learn and understand the basic concepts of AWS Services and how it can be implemented for the betterment of any industry.

References:

[1] https://proquest.safaribooksonline.com/book/web-development/web-services/9781617294440

[2] https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#getting-started-prerequisites

[3] https://aws.amazon.com/premiumsupport/trustedadvisor/

[4] https://aws.amazon.com/cloudformation/

[5] https://aws.amazon.com/s3/

[6] https://aws.amazon.com/iam/

[7] https://aws.amazon.com/ec2/

[8] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html