IOP Conference Series:
Materials Science and
Engineering

PURPOSE-LED
PUBLISHING™

PAPER • OPEN ACCESS

Highly scalable and load balanced web server on AWS cloud

To cite this article: M Mangayarkarasi *et al* 2021 *IOP Conf. Ser.: Mater. Sci. Eng.* **1055** 012113

View the article online for updates and enhancements.

Highly scalable and load balanced web server on AWS cloud

M Mangayarkarasi [1*], S Tamil Selvan [2], R Kuppuchamy [3], S Shanthi [4], S R Prem [5]

[1] Assistant Professor, Department of CT-PG, Kongu Engineering College, Perundurai
[2] Assistant Professor, Department of CSE, Erode Sengunthar Engineering College, Perundurai
[3] Professor, Department of MCA, PSNA College of Engineering and Technology, Dindigul
[4] Associate Professor, Department of CSE, Kongu Engineering College, Perundurai
[5]Student, Department of CT-PG, Kongu Engineering College, Perundurai

*E-mail: vmmangai@gmail.com

Abstract- Most of the industries maintaining their web server on their on-premise, which leads them to have high maintenance, more workload, increases cost. To overcome this we can use cloud technologies for building a web server which gives better performance. Thus we use AWS cloud technologies. In this research paper, a web server is built using the cloud formation technique CFT. Using this technique we can able to build our infrastructure using YAML code. To achieve these major cloud resources are used such as Ec2 Instance, Application load balancer for balancing the load given by users, auto-scaling group. We will configure these cloud resources as YAML and implement them using the cloud formation technique. So that the industries can have their Infrastructure as Code.

Keywords: EC2, Auto scaling group, yaml, Application load balancer.

1. Introduction

Cloud computing is one of the most emerging techniques in IT industries, where cloud providers have their own servers and use access them. It gives an easy and efficient way to spread IT industry services in the world. Nowadays most companies are enhancing their growth by cloud technologies. Cloud Technologies helps them to have fewer infrastructures with high workloads at low cost. This article mainly focuses on building a web server using Cloud Formation Technique.

2. Literature survey

In this paper [1] the author compares load balancer load balancing techniques for a scalable web server. The author redirects the traffic with different load balancing algorithms. In a conclusion, the author states that the round-robin algorithm if more efficient than others while balancing the load.

A survey [2] about the enhancement of auto-scaling and load balancer the author done a performance analysis of load balancing the own physical server and cloud server. As a result, the author concluded that cloud services givers better performance.

In this paper [3] the authors researched how an existing loads balancing system and Auto Scaling Group impacting on each other. The resources used are ELB, ASG, and Workload. They had concluded that developers want to choose the load balancer based on their workload. By default load balancer performs a round-robin algorithm for load balancing.

The author [4] takes a survey on how the auto-scaling and load balancer feature will impact the industries. The author compared these resources on different cloud platform providers like AWS, Google, and AZURE. The author concludes that every cloud provider is given an effective way of improving technologies on the cloud. Users can pick one cloud provider based on their choices.

In general the author [5] gives a survey about cloud computing. How the industries have been changing before and after cloud computing.

For load balancing the users request the author [6] introduced a new technique Auto Load Balancer [ALB] which gives more performance that other load balancing algorithms.

The author [7] develops a web portal different from traditional approach, as a result the authors concluded that the cloud web portal gives more efficiency that ordinary traditional approach.

3. Ec2 instance with nginx

EC2 [ELASTIC COMPUTE] is one of the major service in AWS. An ec2 Instance is a virtual machine in AWS for running software in the cloud, like Virtual Box, VM Ware. For every instance, while launching there will a Security Group attached to it. This Security Group will act as network rules for our virtual machine [Instances]. We can able to set network Inbound and Outbound rules by defining Port numbers shown in **Figure 1**. Here the EC2 Instances act as a web server which server's website and respond to the user's requests. For displaying a website on the browser we need to install an Nginx application on our server [Instance]. Nginx is a software used for serving the web pages on the browser. Nginx is similar to Apache and Tomcat.

We SSH [Secure Shell] into our instance via port 22 and install the software Nginx using the command **yum install nginx -y** for the type Amazon Linux instances or cent OS Linux instances. For ubuntu type Linux instances the command **apt-get install nginx –y**.

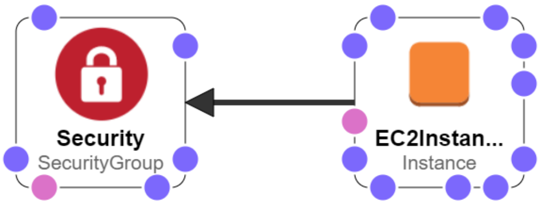


Figure 1. Ec2 Instance with Security Group

4. Elastic file system [efs]

EFS is one of the cloud network storage services given by AWS. It can be used for cloud storage and on-premise resources. It is a dynamic storage service. It is easy to connect, highly scalable, and available among different availability zone. Files can be encrypted while storing in EFS, so it will additional security.

EFS service gives us two performance modes.

1. General Purpose
2. Max I/O

General Purpose is used for normal input and output operations. It is generally used for web serving, content management services, etc. Max I/O is used for big data, machine learning which has a high performance of input and output operations.

There is another feature in EFS which is Infrequently Access Storage Class [EFS IA]. In which we can store the infrequently accessed files, to reduce the cost. We can also enable Simple EFS Lifecycle Management. In which we can set certain files that can move to EFS IA after a certain period. Which will reduce the user time and it will be dynamic.

Here we use the EFS to store our websites and Nginx will serve the web pages form EFS. For connecting EFS with our server first we should install NFS [Network File System] in our server. After installation, next wants to create a directory which will be the mount point for EFS. The command used to mount EFS for Linux servers is **sudo mount -t nfs -o nfsvers=4.1, rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport mount-target-DNS:/ ~/efs-mount-point. Figure 2**. shows the Ec2 [server] mounted with EFS.

For example, if we have many servers hosting a website, if the developers want to do some changes to the website, they have to do changes to all the servers available, it will take more time and more workload. Since we are using EFS so all the servers are mounted, if they do changes to one server it will automatically reflect all the servers because EFS act as common storage for all the servers.

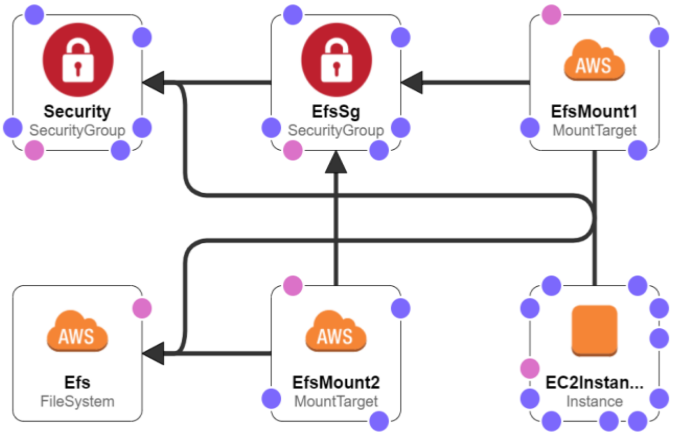


Figure 2. EFS mounted with EC2

5. Load balancer and target groups

Load Balancer in one of the important features should be used for building a web server. Load Balancer is used for balancing the load given by the users. It will distribute the load among different servers. We can able to distribute the load to other availability zone servers, so that if one server at one zone gets down the other server at another zone can able to withstand the load. It makes our server highly available among different zones.

There are three types of load balances in AWS Cloud

1. Network Load Balancer
2. Application Load Balancer

3. Classic Load Balancer

NETWORK LOAD BALANCER - It is useful for transferring the data's at TCP and UDP level. It works under Layer 4. It can able to handle millions of requests per second. For TCP it uses the Hash algorithm for balancing the request based on source IP, port number, and destination IP. We can able to enable health checks for our server through load balancing. If sever does not respond the health check responds with 504 error.
APPLICATION LOAD BALANCER – It is useful for balancing the HTTP and HTTPS traffics. It works under layer 7. It transfers the requests independently; it does not use any algorithms. Here the routing is fully based on Target Groups.
CLASSIC LOAD BALANCER – it is the oldest load balancer in AWS. However, AWS will not recommend using this load balancer. It performs a basic load balancing among servers.

5.1 Target groups

Every load balancer will be attached to Target Groups. Target Groups tells the load balancer where to send the traffic to which server. Every server should get registered to the Target Groups because at first, the load balancer checks the target group, then it checks the registered server in it and it can able to distribute the load among only the registered server. If a server is non-registered, then it becomes idle the load will not distribute in it. A health check will automatically enable, because if one server gets down it tells the load balancer to send the traffic to a healthy server. While creating a target group by default a Target Group Listener will attach to it. Here the listener acts as a rule for the routing.

Developers can able to use this for more advanced routing by defining port, IP address. Further creating servers can be able to add to target groups for load balancing without interrupting the other server and can able to de-register if the server gets down. By default, Target Group uses the Round Robin algorithm for routing the traffic among the targets [servers]. More than one listener can be able to add to a Target Group. Check **Figure 3.** in which the load balancer is attached to a target group with a listener and the EC2 Instance [server] are registered to the Target Group.

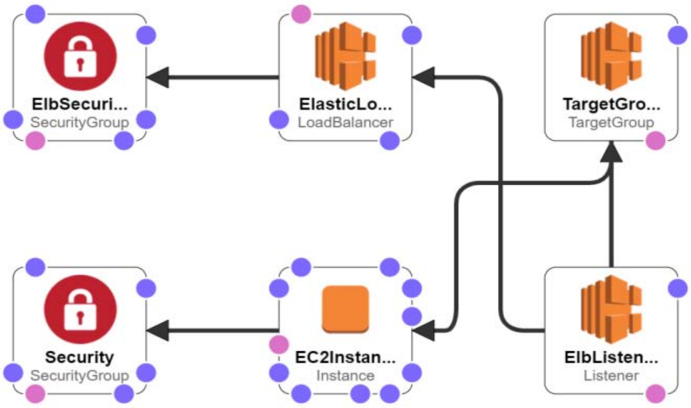


Figure 3. Load Balancer and Target Groups

6. Auto scaling group [asg]

Auto Scaling Group is used for automatic scaling and managing the servers. ASG can able to create a new server or terminate the server based on the given scaling policies, so the server becomes dynamically scalable and highly available all the time. ASG launches a new server based on the launch template or launch configurations attached to it.

For example, if one server gets down at one zone, ASG will check the status and it terminates the unhealthy server and launches a new one at that zone.

We can able to fix the minimum, maximum, and desired capacity of the server. While the load gets high the ASG will launch the new servers up to the maximum number that is given. If the load gets lesser it will automatically terminate the idle server, by using ASG it reduces workload and can save more cost.

6.1 Scaling policies

Scaling Policies are attached with the ASG, it tells the ASG when to launch a new server and when to terminate the idle server based on the scaling policies given.

Types of scaling policies

1. Target Tracking Scaling
2. Step Scaling
3. Simple Scaling

6.2 Launch configuration

It is a server configuration templates used in ASG. In the launch configuration, we can able to specify the server type and what is the software needed to be installed before launching.

While creating an ASG a launch configuration must be created and attached to it, because with the help of launch configuration the ASG can able to launch a server with the configurations specified in it.

Figure 4. shows that ASG is attached to Launch Configuration and Scaling Policies. Here for the web server, we create a launch configuration with nginx software installed in it for serving the web pages on the browser.

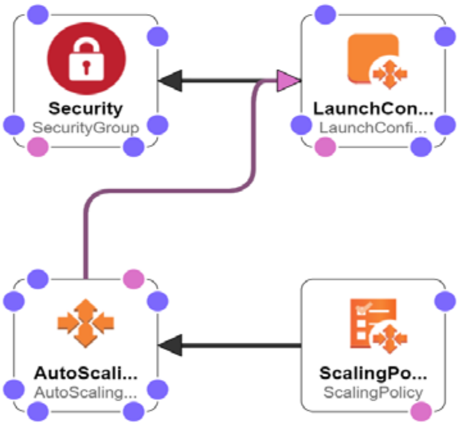


Figure 4. Auto Scaling Group

7. Virtual private cloud [VPC]

VPC is used for network configurations in AWS Cloud. We can able to create our VPC in the Cloud. Every resource in AWS must deploy in a VPC. By default, AWS gives us a Default VPC connection.

VPC major Components are

1. Subnet
2. Route Table
3. Internet Gateway
4. NAT Gateway

SUBNET - is one of the components in a VPC. Default VPC contains public subnets in which anyone can able to access it, but we can able to create a new subnet as private so that only the specified IP addresses can able to access the server. It makes the server more secure.

ROUTE TABLE – It is used in VPC for routing the traffic. The route table consists of certain rules which determine which traffic should be redirected to which subnets either private or public.

INTERNET GATEWAY – it establishes a connection between our cloud VPC and the Internet. One VPC can only have One Internet Gateway attached to it. It is mostly recommended for public subnets.

NAT GATEWAY – It establishes a connection between private subnets and the Internet. By default, the Public subnets reach the Internet Gateway for internet connection and the private subnet reaches the NAT Gateway. Nat Gateway uses an Elastic IP Address for establishing a connection, so an Elastic IP will get associated automatically with the NAT Gateway while creation.

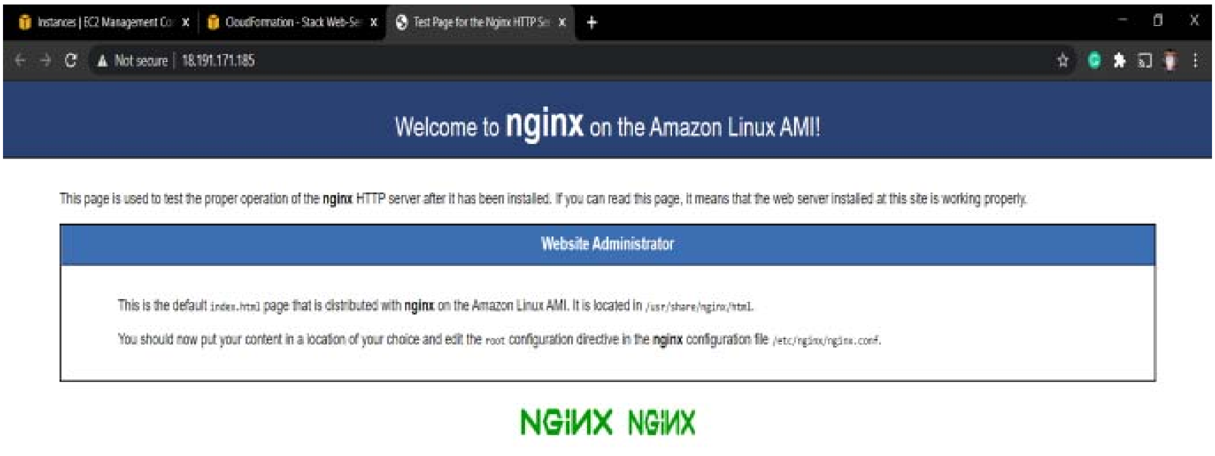
Here we deploy our web server in private subnets so that it will be more secure. The load balancer and Auto Scaling Group should be in the public subnet for users so that they can able to access our server through the load balancer DNS name.

8. Cloud formation

Cloud Formation Technique [CFT] is used for building the INFRASTRUCTURE AS CODE refer [8]. It uses JSON or YAML code for building the infrastructure. The proposed system of this paper is by building the whole web server infrastructure using CFT.CFT works by uploading a template to it. The template can be created by JSON or YAML code or by python package called TROPOSHERE. The advantages of using this technique are that developers can easily manage their infrastructure as code, reduce work time and workload, code reusability.

9. Results

The Web server has been built successfully using YAML code in the cloud formation technique. **Figure 5**. shows that the server has been successfully installed and the Nginx startup page is displayed in the browser.



Instances | EC2 Management Co
CloudFormation - Stack Web-Se
Test Page for the Nginx HTTP Se
Not secure | 18.191.171.185

Welcome to **nginx** on the Amazon Linux AMI!

This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

Website Administrator

This is the default `index.html` page that is distributed with **nginx** on the Amazon Linux AMI. It is located in `/usr/share/nginx/html`.

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

Figure 5. Server startup page

Figure 6. Says about the total amount of requests that are responded by the server. The green line indicates the successful requests and the red line indicates the delayed requests. X – axis gives the time at which the server responded. Y – axis says the number of request / sec reached the server. When the number of request increases there will be a slight delay in response by server and it can be increased by increasing the server RAM, CPU, GPU capacity.

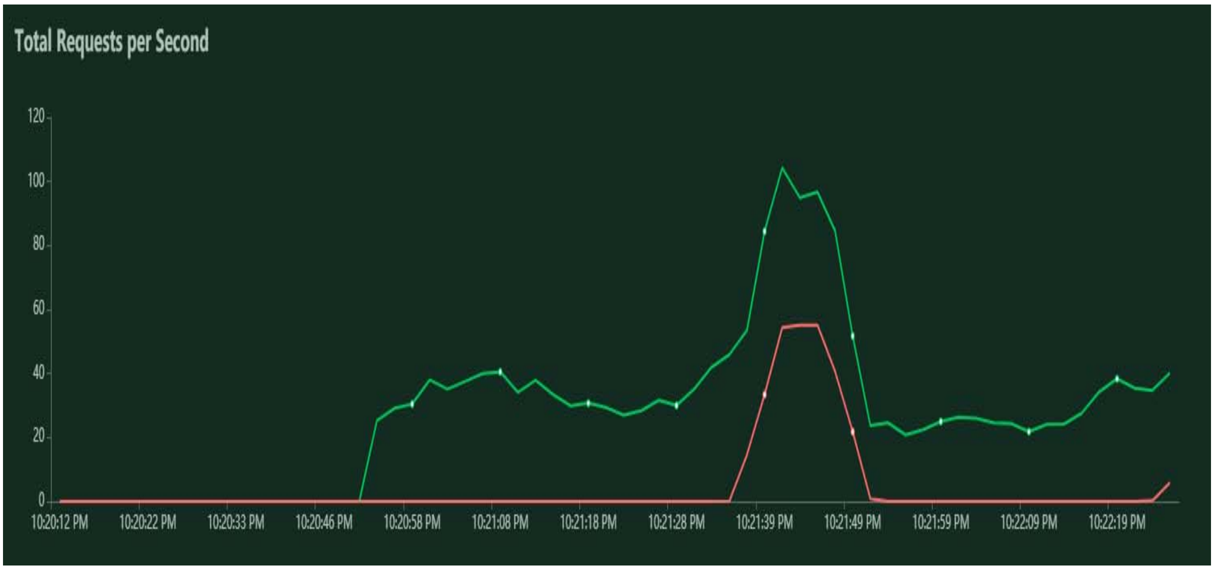
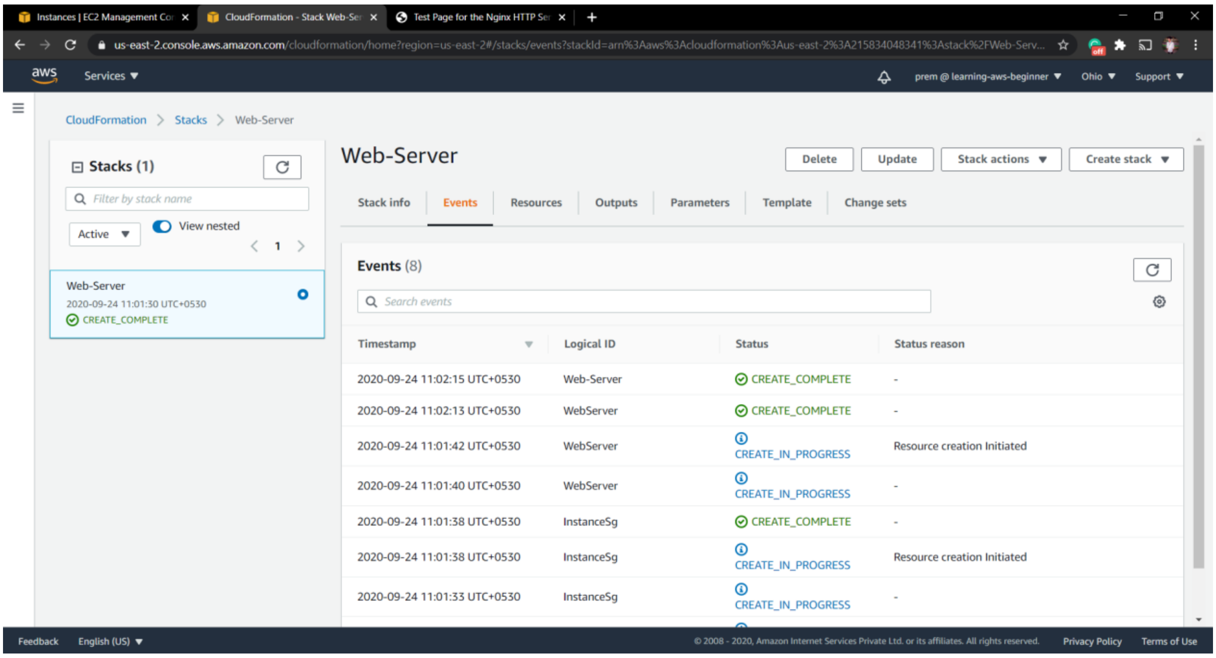


Figure 6. Total Request per second

🟩 - Successful Requests

🟧 - Delayed Requests


Instances | EC2 Management Cor
CloudFormation - Stack Web-Ser
Test Page for the Nginx HTTP Ser
us-east-2.console.aws.amazon.com/cloudformation/home?region=us-east-2#/stacks/events?stackId=arn%3Aaws%3Acloudformation%3Aus-east-2%3A215834048341%3Astack%2FWeb-Serv...
aws
Services ▼
prem @ learning-aws-beginner ▼
Ohio ▼
Support ▼
CloudFormation > Stacks > Web-Server
Stacks (1)
Filter by stack name
Active ▼
View nested
1
Delete
Update
Stack actions ▼
Create stack ▼
Stack info
Events
Resources
Outputs
Parameters
Template
Change sets
Search events
Feedback
English (US) ▼
© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Web-Server
2020-09-24 11:01:30 UTC+0530
✓ CREATE_COMPLETE

Web-Server

Events (8)

Timestamp	Logical ID	Status	Status reason
2020-09-24 11:02:15 UTC+0530	Web-Server	✓ CREATE_COMPLETE	-
2020-09-24 11:02:13 UTC+0530	WebServer	✓ CREATE_COMPLETE	-
2020-09-24 11:01:42 UTC+0530	WebServer	ⓘ CREATE_IN_PROGRESS	Resource creation Initiated
2020-09-24 11:01:40 UTC+0530	WebServer	ⓘ CREATE_IN_PROGRESS	-
2020-09-24 11:01:38 UTC+0530	InstanceSg	✓ CREATE_COMPLETE	-
2020-09-24 11:01:38 UTC+0530	InstanceSg	ⓘ CREATE_IN_PROGRESS	Resource creation Initiated
2020-09-24 11:01:33 UTC+0530	InstanceSg	ⓘ CREATE_IN_PROGRESS	-

Figure 7. Cloud Formation

Figure 7. shows the output screen of the Cloud formation Technique. It shows that that the web server has been created successfully using Cloud Formation Technique [CFT] and YAML code in it and the logical id of our cloud resources used for building web server. Cloud formation technique creates stack for the code and it will maintain our infrastructure. There is an update option in stack in which the infrastructure can able to upgrade to new version. The main advantage by using cloud formation technique is **CODE REUSABILITY.**

10. Conclusion

This work is focused on building the infrastructure as code. There are lots of cloud resources configured in the code such as ec2 instance, application load balancer, auto-scaling group. By using the infrastructure as code saves a lot of workloads, time, and code reusability. We used locust for testing the webserver for handling the load shown in Figure 6. The result shows that by using cloud technologies the webserver gives higher performance. This word concludes that we have the whole infrastructure as code which leads the industries to have more advantages than on-premise web servers.

References

[1] Haakon bryhni 2000, A Comparison of Load Balancing Techniques for Scalable Web Server, *Article at IEEE Network.*

[2] Ankit kumar 2019, Enhancement of Auto Scaling and Load Balancing using AWS, *SKIT Research Journal,* **9**

[3] Nguyen hong Son 2017, Load balancing in auto scaling-enabled cloud environments, *International Journal on Cloud Computing: Services and Architecture* **7**

[4] Ashalatha R, Jayashree agarkhed 2015, Evaluation of Auto Scaling and Load Balancing Features in Cloud, *International Journal of Computer Applications* **117** 0975 – 8887.

[5] Priyanshu srivastava, Rizwan khan 2018, A Review Paper on Cloud Computing, *International Journals of Advanced Research in Computer Science and Software Engineering.* **8**

[6] Arvindhan M 2019, Scheming an Proficient Auto Scaling Technique for Minimizing Response Time in Load Balancing on Amazon AWS Cloud, *International Conference on Advances in Engineering Science Management & Technology 2019*

[7] Prashant tyagi, Satyam singh, Ravi prakash chaudhary, Praveen kumar singh 2020, Building Web Application using Cloud Computing, *International Research Journal of Engineering and Technology* **7**

[8] Gurudatt kulkarni, Ramesh sutar, jayant Gambir 2012, Cloud computing infrastructure service amzon ec2*, International Journal of engineering research and application* **2** 117-125