Capstone Project Weekly Progress Report

Project Title	**Zene Cloud**
Group Name	**Group-I**
Student names/Student IDs	**Charanjeet Singh (C0740039)** **Hemal Chudasama (C0734999)** **Kanwaldeep Singh (C0735948)** **Surbhi (C0736237)**
Reporting Week	**10 June 2019 – 16 June 2019 (Week 5)**
Faculty Supervisor	**William Pourmajidi**

1. Tasks Outlined in Previous Weekly Progress Report

- We have created ELB (Elastic Load Balancer)
- Then we configured Launch Configuration and AutoScaling
- We then enabled ACM (AWS Certificate Manager) and ELB (Elastic Load Balancer)
- We secured infra at Security Group and VPC level.
- We then tested by terminating all Instances. To check that AutoScaling should be able to create new EC2 instances.

2. Progress Made in Reporting Week

- Creating Elastic Load Balancer, following are the steps:
  - Create Load Balancer
  - Create Application Load Balancer

    1. Configure Load Balancer

       ✓ Name your Load Balancer
       ✓ Select scheme as "Internet Facing"
       ✓ Select Ip address type as "Ipv4"
       ✓ Select Load Balancer protocol "HTTP"
       ✓ Select Load Balancer port "80"

- ✓ Select the Availability Zone you want to re-direct the traffic to.

2. Configure Security Group

    - ✓ Create new Security group using TCP protocol on port 80 which allows traffic from anywhere (0.0.0.0/0, ::/0)

3. Configure Routing

    - ✓ Select Target group where you want to re-direct the traffic
    - ✓ Select Health Check protocol as HTTP.

4. Register Target Group

    - ✓ Select all the EC2 instances you want to target

- o Click on Create Load Balancer

NOTE:
We chose Application Load Balancer instead of Network Load Balancer because Application Load Balancer (ALB) provides routing features using HTTP/HTTPS whereas NLB is mainly used to provide ultra-high performance to re-direct TCP/TLS protocol. Application Load Balancer (ALB) is more suitable for our project because it provides flexible feature set as compared to Network Load Balancer (NLB).

- Creating Launch Configuration and Auto Scaling
    - o Click on Create Launch Configuration
    - o Choose Amazon Machine Image (AMI)
    - o Choose Instance Type
    - o Configure details
        - ✓ Add Name
        - ✓ Select IAM role, if any
        - ✓ Click on Advance Details to add User Data

The following are the command to install Apache Server using EC2 User Data

```
#sudo su
#yum update -y
#yum install -y httpd
#chkconfig httpd on
```

- o Configure Security Group
    - ✓ Add SSH rule on Port 22
    - ✓ Add HTTP rule on Port 80
    - ✓ Add HTTPs rule on Port 443

NOTE:
We have added HTTP/ HTTPS rule because Application Load Balancer (ALB) re-directs HTTP/ HTTPS traffic.

  - Select a Key Pair
  - Click on Create Launch Configuration

- Configure Auto Scaling Group Details
  - Add Group Name
  - Select VPC network
  - Select Subnets

- Advanced Details
  - Tick the Load Balancing check box
  - Select the target group
  - Select Health Check type as ELB (Elastic Load Balancer)

- Configure Scaling Policy

- Click on Create Auto Scaling Group

NOTE:
We created a Load Balancer because it evenly distributes the traffic to the EC2 instances so that a single server will never get all the requests. We enabled the feature of Auto Scaling amongst our EC2 instances to make our web app highly available all the time. In case, an instance fails ASG (Auto Scaling Group) will automatically create a new instance.

3. Difficulties Encountered in Reporting Week

- While creating Auto Scaling group (ASG), we kept the default cooldown period as 300 seconds which is by default and we cannot reduce it. This cooldown period took too much time to place next Scale-in or Scale-out event in effect. The more the cooldown period, the more it costed us. Then we found that out we can add Scaling-Specific cooldown period to our Scaling Policy, which can reduce the default cooldown period.
  - Then we set our cooldown period to 180 seconds, which allowed to quickly place next scaling event and also helped us to reduce the cost.

  `https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html

- In Monitoring Tab of our Auto Scaling Group, we found out that one of our instances was getting Over-Loaded by 80% of its capacity and rest two were still on 20% of its capacity. We could not find out what was the problem. Later then we found out that we enabled Load Balancer Stickiness which kept user's session data intact and every time he gets re-directed to the same server. https://aws.amazon.com/blogs/aws/new-elastic-load-balancing-feature-sticky-sessions/

- Then we disabled Load Balancer Stickiness to avoid the problem.

4. Tasks to Be Completed in Next Week

- We will Create a Relational Database (RDS) backup retention of 5 Days
  - Backup window at 10pm every day.
- Auto upgrade minor versions of Relational Database (RDS)
  - Maintenance window at Sunday 11pm.
- We will create our webpage www.bluehost.com
  - We will be designing a layout; we will buy a template (most probably).
- We will buy a domain.
- We will then configure AWS Route53 with that domain.