Migration

Apache Web Server, Apache Tomcat application server with Active MQ and Oracle and MongoDB backend.

Propose a solution to migrate this application stack to AWS. Mention all the AWS services you would use and how you would maintain HA and Load Balancing (consider app to be stateless). Mention rationale for each design decision.

**Answer:**

AWS Services that we will used:

1) For Apache Web Server, Apache Tomcat application we use EC2 instances

2) Active MQ will be replace with the SQS services in AWS

3) For Oracle, we have to you RDS with BYOL(Bring your Own License)

4) MongoDB will be installed on EC2 instances with Mongo Replication (i.e. min three Ec2 instances)

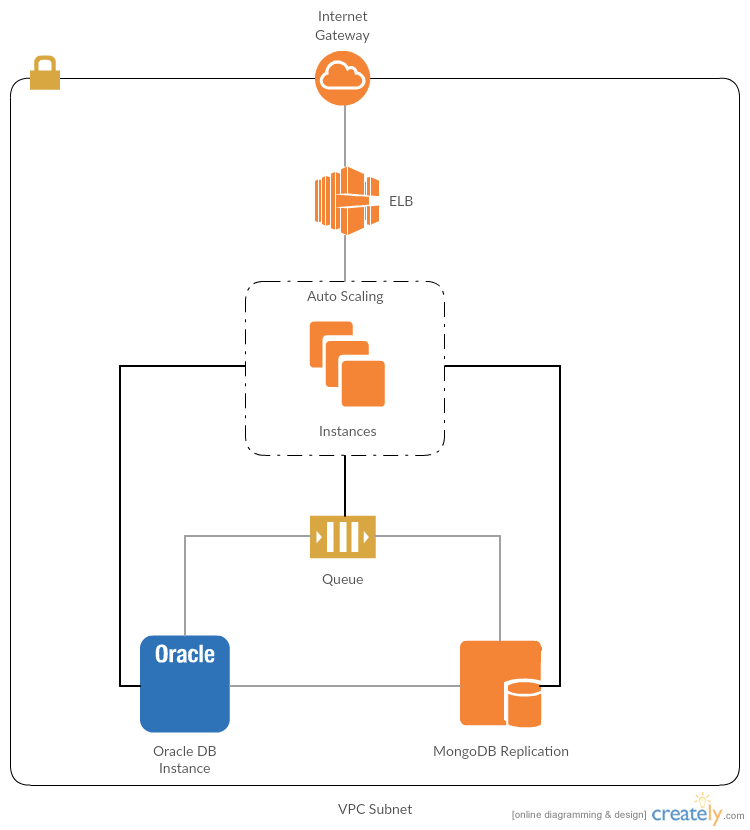
5) All these services will be under once VPC(Virtual Private Cloud) and under ELB(Elastic Load Balancer)

6) Only the ELB will be facing the Public Network, rest all the communication will be held using Private IPs, this can be achieve using Security Groups

7) Also, we need to set Auto Scaling for our Application Ec2 instance, so that they will scale horizontally by adding more compute power when needed

8) Need to use S3 for files

9) We need to use AWS DMS (Database Migration Service) to migrate the Database



Migration:

Migration is a very tedious task. Before we plan to migrate the infra, we need to do the POC(Proof of Concept) for the same.

For this we need to create same stack in the AWS with minimum requirement and test the same. Like one Ec2 install behind the ELB and some test(sample) database.

If everything went fine and you want to migrate the Prod Infra, then we need to do it in 2 Phase:

Phase1:

Create complete infra structure on AWS i.e

1. Create an ELB
2. Take an AMI of instance used in the test environment and use it as base image.
3. Using the base AMI launch the instances in AutoScaling
4. Create MongoDB replication using atleast three EC2 instances
5. Migrate the Oracle database to the AWS, this make take time depending upon the size.
6. We can use the feature of AWS DMS to migrate the database
7. Once all done, test and verify the same

Phase2:

1. Once all verified, take the incremental backup of Database and import that on existing AWS Oracle database
2. We need to schedule the downtime for the same
3. Also, during the schedule downtime, we can also change the DNS pointing to the new infra.
4. Downtime, should be minimum as we have to only migrate the incremental database sync with the producing environment

Failover:

As our old infra is still working , if anything goes wrong, we need to change the DNS pointing to the old infra.