**Assignment 3 :**

Consider a monolithic java application stack.

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.

Explicitly state the assumptions you are making, if any.

**I have assumed HLA Diagram of the application stack as follows -**

MongoDB

**Apache Web Server 1**

**Application Server**

**Reverse Proxy and Load Balancer**

**ActiveMQ Server**

**Browser**

**Apache Web Server 2**

Oracle

**Proposed Solution for Migration**

**Like any other transformations, migration also should go through phases. roughly described those are as follows.**

**Planning** – It will include financial assessment (TOC), technical assessment, prototype creation or proof of concept, data migration planning and application migration planning.

**Execution** – It will have execution of planning, modification of application source code and testing.

**Deployment** – It will have deploy application and data migration.

**Monitor**  – AWS Cloud has AWS CloudTrail and Amazon CloudWatch services for monitor AWS resources.

**Here I am considering following things:**

**-** The application is simple e-commerce application

- The application stack is not going to change after migration. Considering that we don't want to do any change in development as of now.

That means , Servers are not going to replace with any Amazon Web Service.

- I am considering total 7 VMs are required.

1 - reverse proxy server and same as load balancer

2 - apache web servers

1- ActiveMQ Server for messaging purpose

1- Application Server

2 - DB Servers as mentioned in problem statement

**Step 1 - Amazon Management Console**

Using management console you can login to your AWS account and manage your resources.

Login first and then access EC2 Dashboard

**Step 2 - Amazon EC2 Service**

Using Amazon EC2 Service we need to first create these 7 VMs as per pre-known configurations and then we have to test integration between them.

**Step 3 - Configure Networking and security groups**

You have to configure proper network settings and security groups for each and every server so that they all will be able to communicate each other without any impediment or security concern.

**Step 4- Install and Configure each server using automated scripts.**

You should have automated scripts to setup fully functional servers, Using which you can later setup your environment easily anywhere.

NOTE: You can do this manually as well.

**Step 5 - Test Integration Between machines and Deploy your web application**

Once servers are ready, you can deploy your web application on intended application server.

and test it.

**Following web services by Amazon can be used in your application stack, but for that you might need to so some development changes.**

**Amazon S3 Bucket** (Simple Storage Service) - Amazon S3 provides a highly durable and available store for a variety of content, ranging from web applications to media files.

**ELB** (Elastic Load Balancer) - Elastic Load Balancing automatically distributes incoming application traffic across multiple Amazon EC2 instances.

**Route 53** (DNS Service) - Amazon Route 53 is a highly available and scalable Domain Name System (DNS).

**Amazon Simple Queue Service (SQS)** - Instead of ActiveMQ you can use this.