Final Project Report
On
iCare Health Service on Cloud

CSYE 6225
Network Structures and Cloud Computing

Guided By:
Prof. Julian Cooper

Team Name: RedHat

Team Members:

Reshmi Padaval
Bhavna Menghrajani
Gokul Ramakrishnan
Priyanka Singh
Sreenivas Vattikundala

Contents

1. Introduction

- Cloud computing is a general term used to refer to the delivery of hosted services over the internet.

- Cloud computing enables companies to consume a compute resource, such as a virtual machine (VM), storage or an application, as a utility -- just like electricity – without having to build and maintain computing infrastructures in-house.

- Cloud computing boasts several attractive benefits for businesses and end users. Three of the main benefits of cloud computing are:

- Self-Service Provisioning: End users can spin up compute resources for almost any type of workload on demand. This eliminates the traditional need for IT administrators to provision and manage compute resources.

- Elasticity: Companies can scale up as computing needs increase and scale down again as demands decrease. This eliminates the need for massive investments in local infrastructure which may or may not remain active.

- Pay per use: Compute resources are measured at a granular level, allowing users to pay only for the resources and workloads they use.

2. Requirements

- Develop a login portal using Java Spring MVC which will consist of use-case buttons to simulate an increasing load on the application and database.
- The database can either be a relational database or a NoSQL database.
- The entire stack must be deployed in AWS or Azure.
- Proper infrastructure alerts and triggers to allow for auto-scaling of resources to accommodate the additional load in application, network, data storage and usage with your environment.
- Minimum of one load balancer is required.
- There are 5 use-case buttons that need to be implemented which are follows:
    - Simulation of 3 users log in to application and run 2 report going back 10 days
    - Simulation of 10 users log in to application and run 6 report going back 30 days
    - Simulation of 17 users log in to application and run 10 report going back 60 days
    - Simulation of 24 users log in to application and run 14 report going back 90 days.
    - User input fields to allow me the ability to enter parameters for how many users log in simulation and how many reports those users are uniquely being running.

3. Business Justification

We should deploy a spring/web application on the cloud (Azure / AWS) and the backend database should be an RDS database. Deploying the application in the cloud with the help of an appropriate load balancer and an Auto-scaling scheme should take care of the automatic scaling up and down of the server and should also avoid latency and bottleneck issues.

4. Systems and Descriptions

a. Techniques Used:

- Load Balancing
- Auto Scaling
- Cloud Watch Monitoring

b. Platform:

Amazon Web Services

c. Web Application

We have a spring MVC application and are using MySQL with RDS as Database.

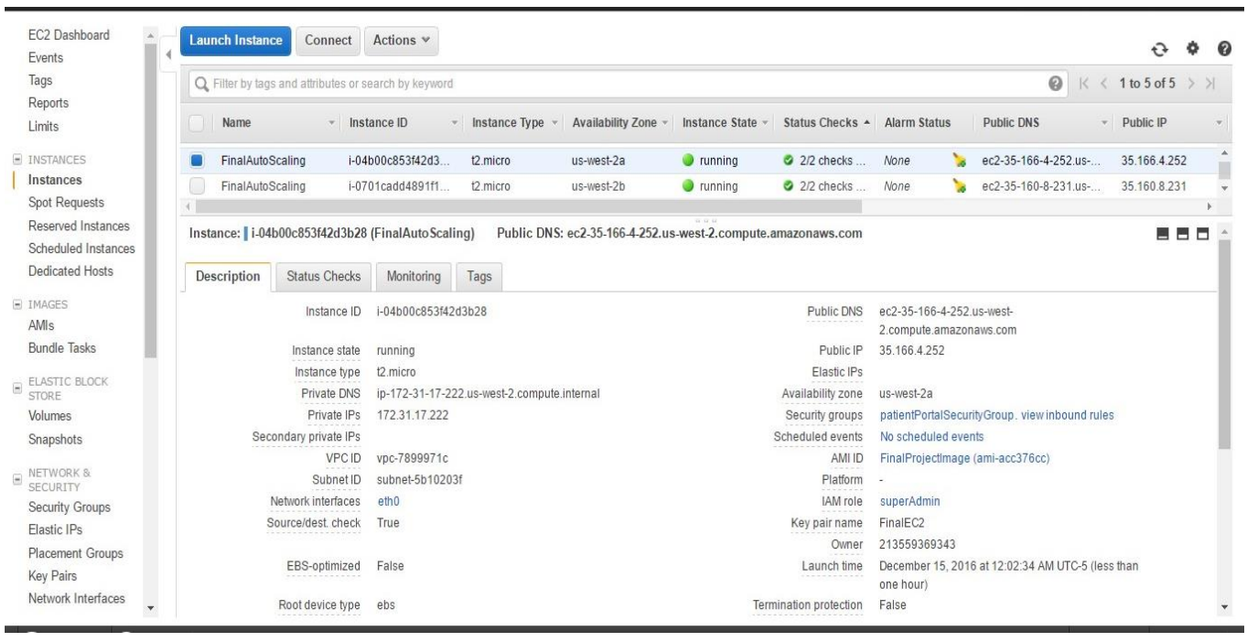
- We have picked a sample dataset from the following URL: https://www.healthdata.gov/search/type/dataset
- The size of this dataset is 5MB and it has nearly 10000 records in the dataset table which contains predominantly patient health related information, which is solely used for report generation. The application has an authentication mechanism and allows users to generate load on the system. The application internally invokes bash script which in turn calls python script to retrieve the records as per use cases.

d. Login Simulation

- We use a python script to trigger user login simulation. The python script is programmed to trigger at the click of use case buttons in the Web application.
- The various use case buttons are preprogrammed with a set value to simulate the logging in of n number of users.
- The python script is triggered using a bash script on AWS.
- Once the python script starts to run, n number of users log in to the web application and try to fetch data from the database which creates a load on the EC2 instance which triggers load balancing and auto scaling.
- The activity on AWS can be observed using Cloud Watch.

e. Launching an instance using private AMI

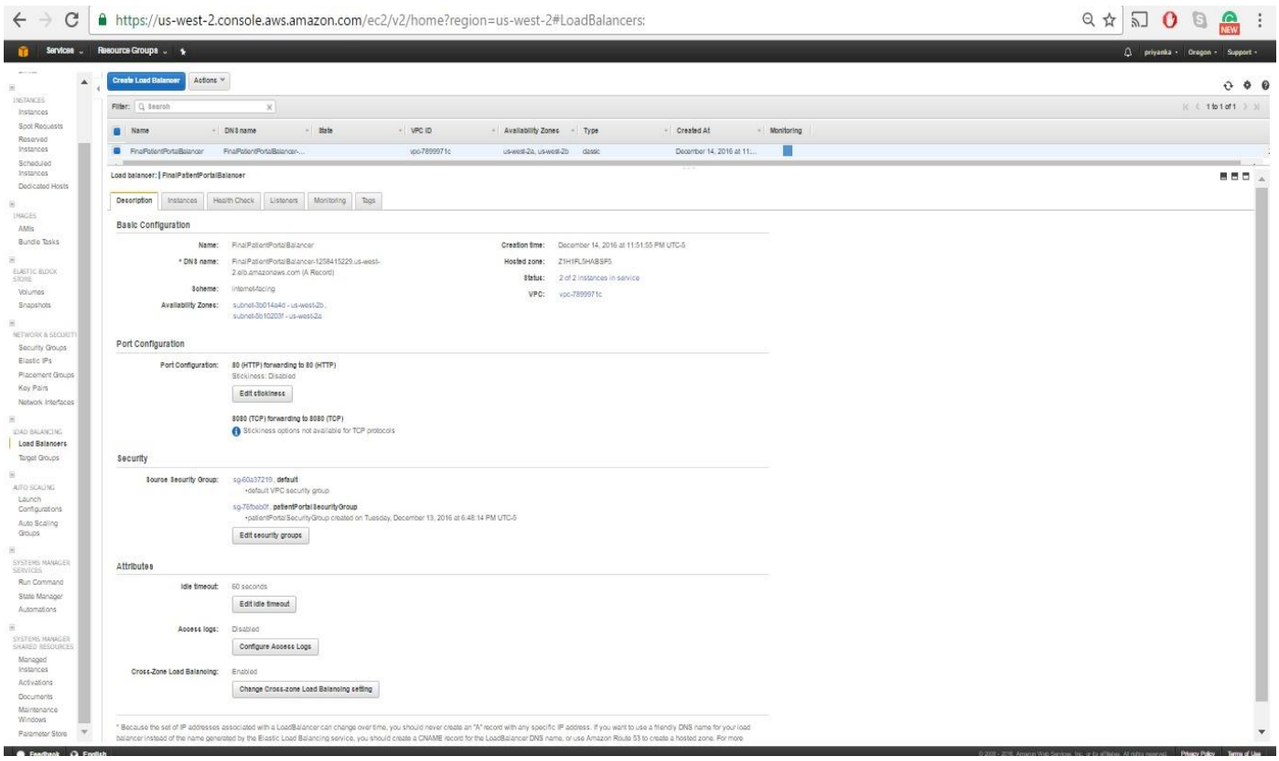
- An AMI contains the information required to create a new instance. For example, an AMI consists of required platform being configured: Apache Tomcat 7, mySQL, python 2.7 and Spring MVC web application. On choosing the private AMI and providing the necessary security groups and to remain eligible for the free tier, we chose the t2.micro instance type. Now the instance is created.



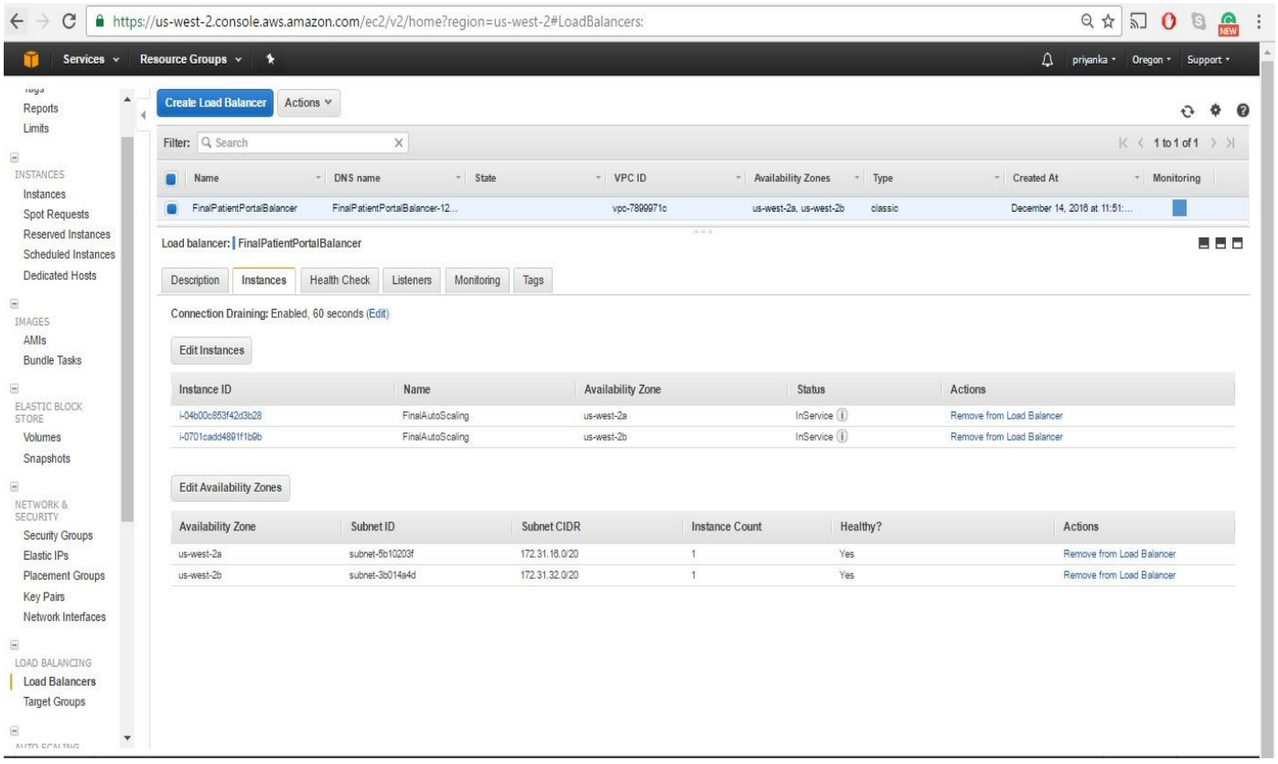
f. Creating Load balancer to listen on http:

- What is load balancing? Load balancing distributes workloads across multiple computing resources, such as computers, a computer cluster, network links, central processing units or disk drives. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. We Click Add Listener to create a new listener for the ELB.
- LB Protocol: We Select 'HTTP'
- LB Port: Enter '8080'
- Instance Protocol: Select 'HTTP'

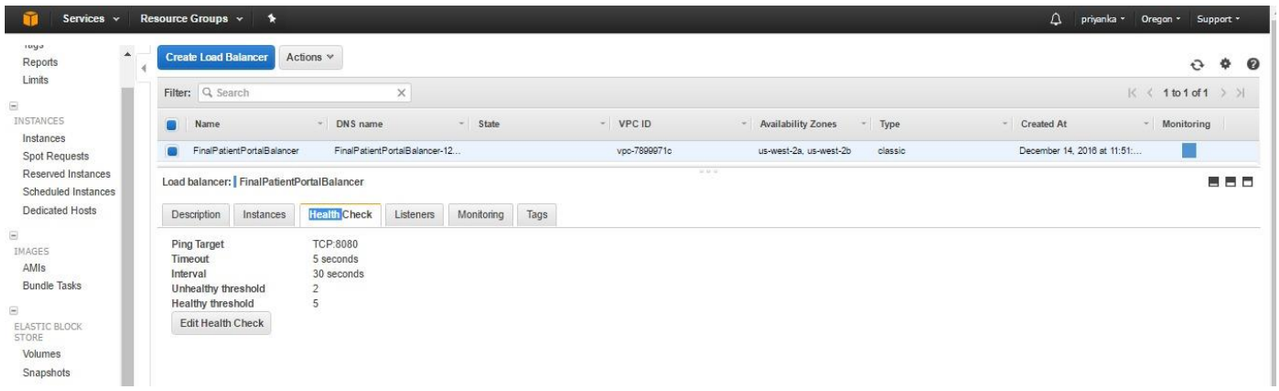
Load Balancer Basic Overview



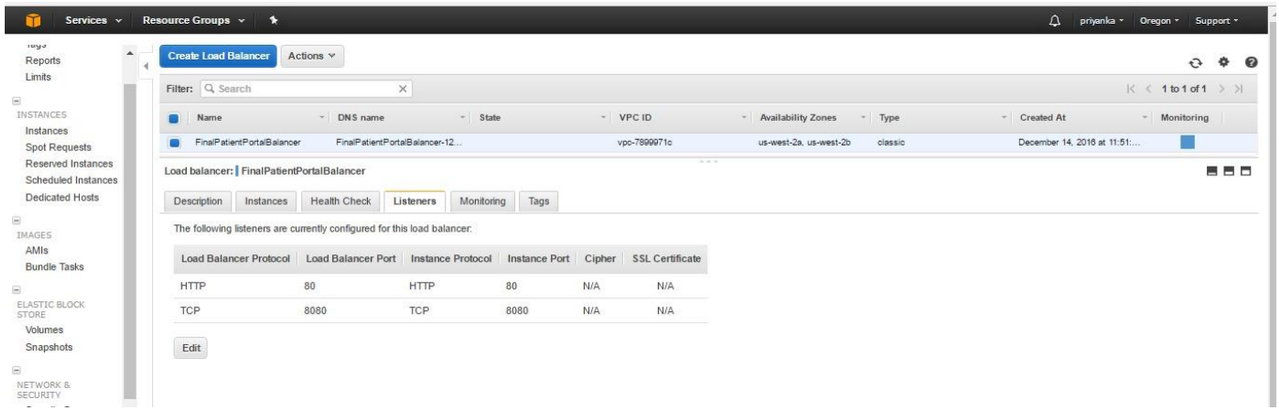
Load Balancer Instance Details



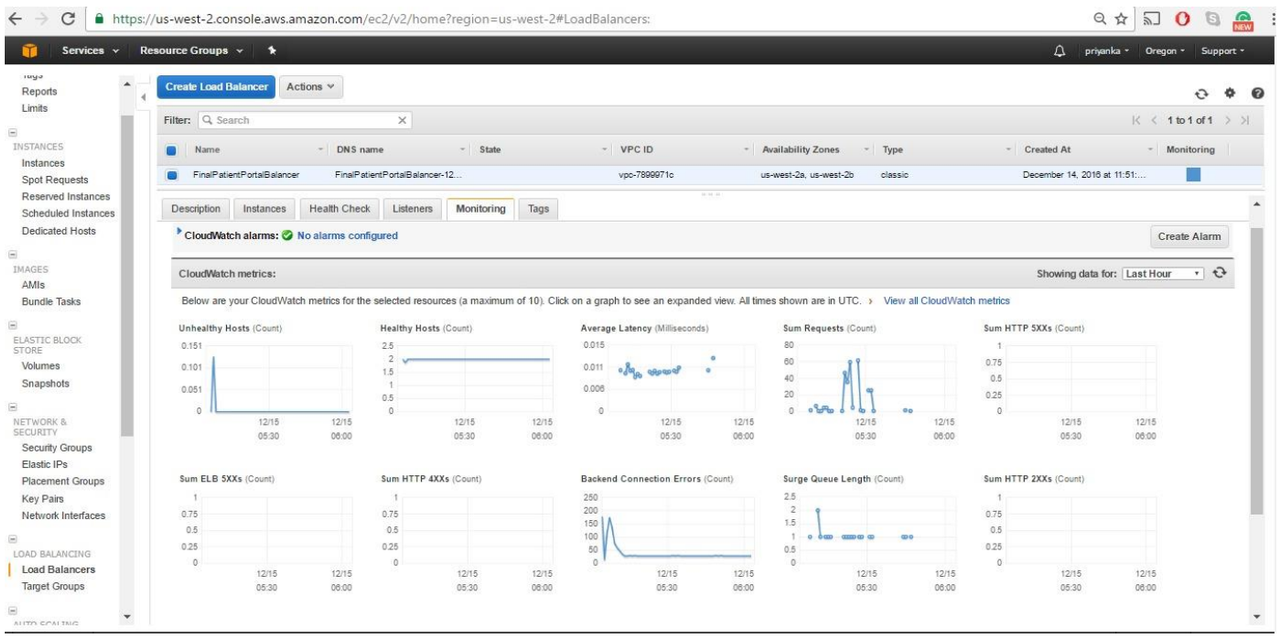
Load Balancer Health Details



Load Balancer Listeners



Load Balancer Monitoring



g. Amazon RDS:

We have chosen MySQL database engine for RDS.
Mainly for 4 reasons.

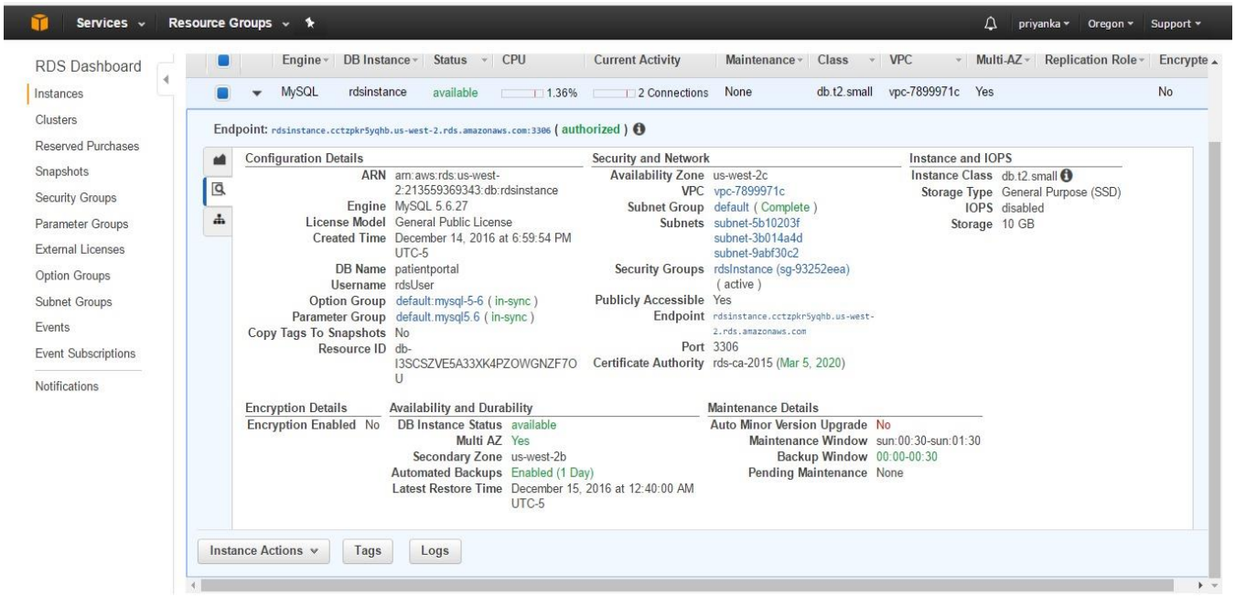
- Scalability and Flexibility
- High Performance
- High availability
- Automatic backups and maintenance Amazon RDS for MySQL gives access to the capabilities of a familiar MySQL database engine.

- Creating a DB Instance:

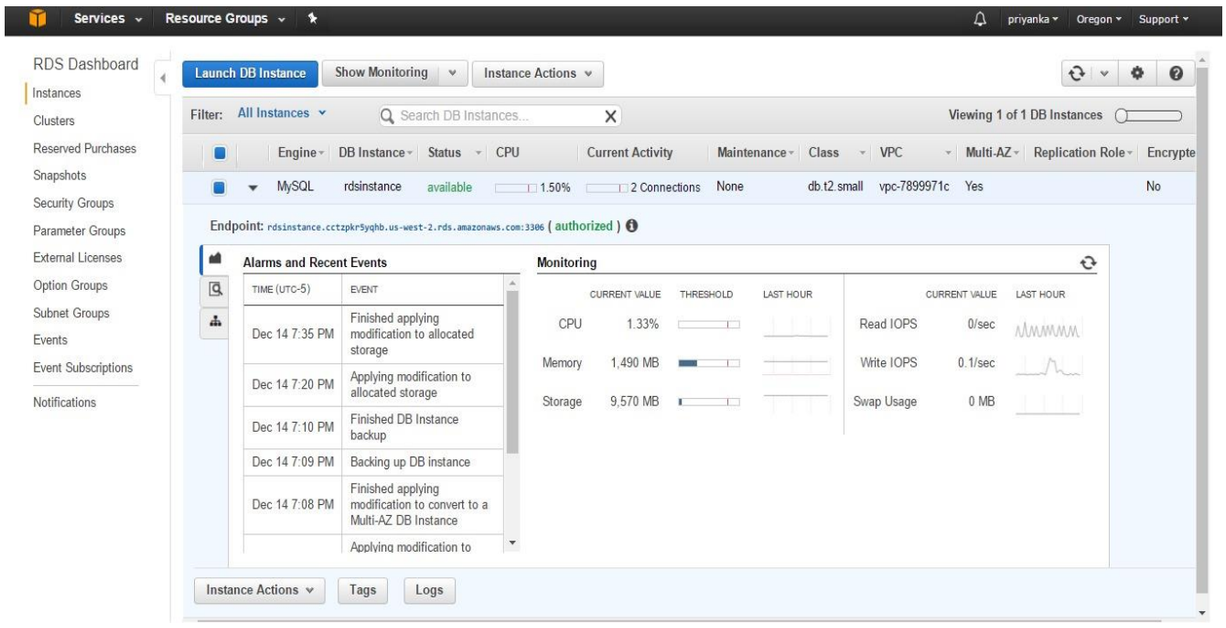
First we create a DB instance. On the Specify DB Details page, specify your DB instance information that includes License model, Db Engine version, DB instance class and DB instance identifier. Then on the Configured Advanced setting page we provide additional information that includes VPC, Subnets and Security groups.

- RDS Connection with Workbench:

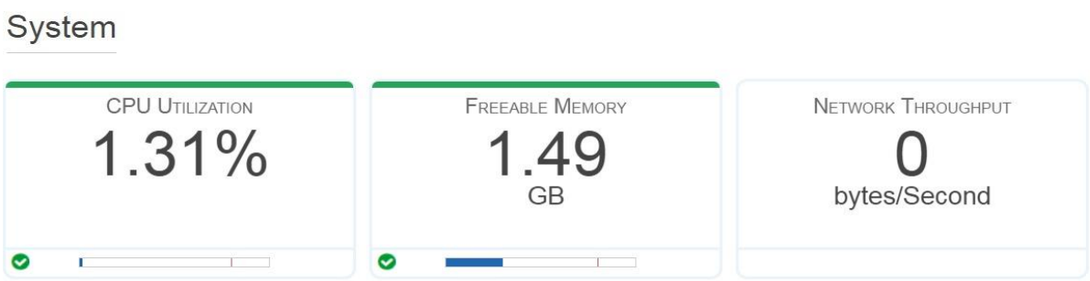
We connect the database instance of RDS with MYSQL Workbench. We specify the endpoint of the DB instance with the username and password that was given during the launching of the DB instance.

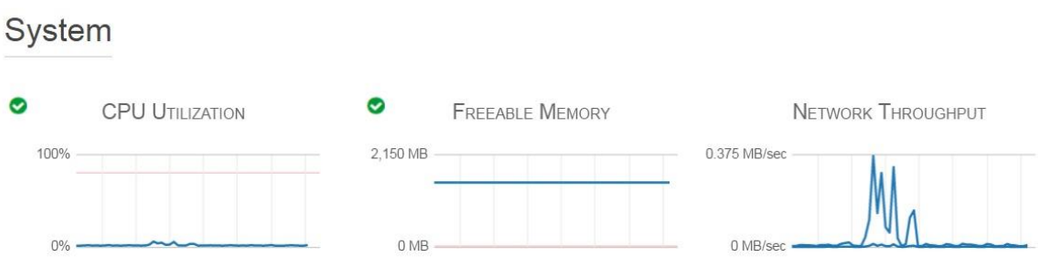


RDS Utilization, Alarms and Recent Events:

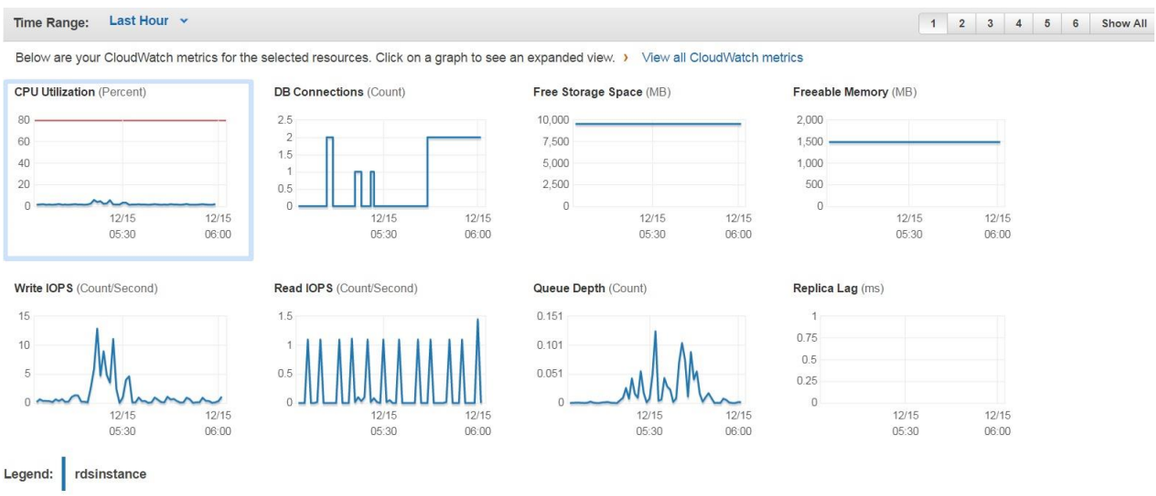


Database Monitoring: Overview





Complete Graph of all monitoring metrics:



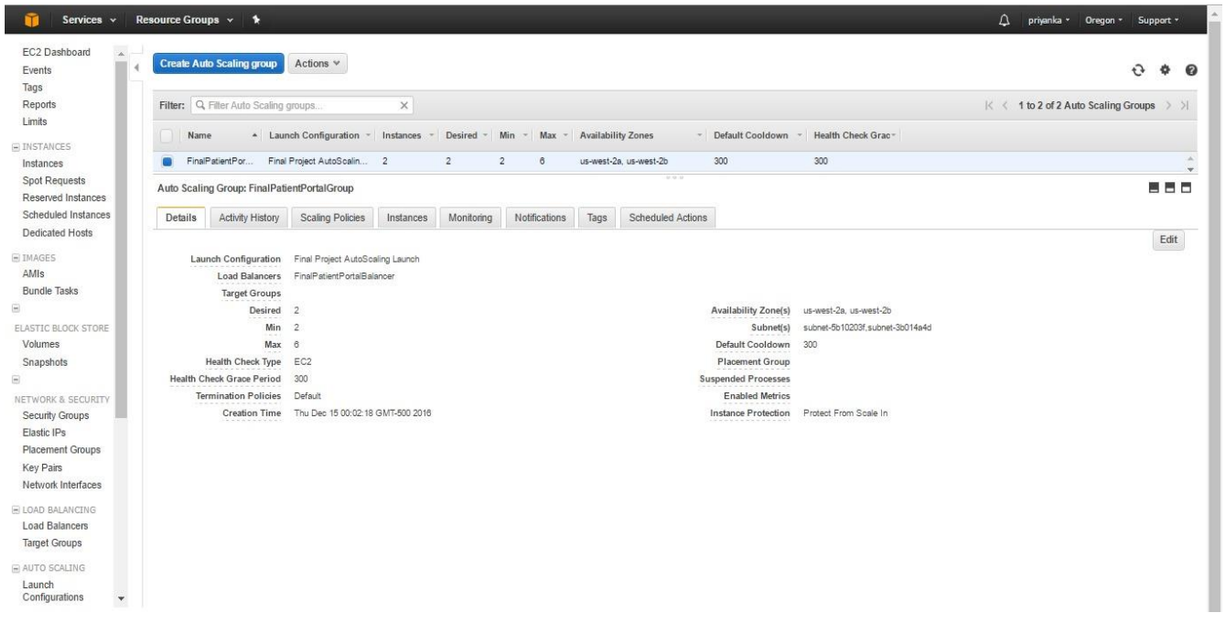
h. Auto Scaling Group:

What is Auto Scaling?
Auto-scaling is a cloud computing service feature that automatically adds or removes compute resources (instances) depending on actual usage demands.

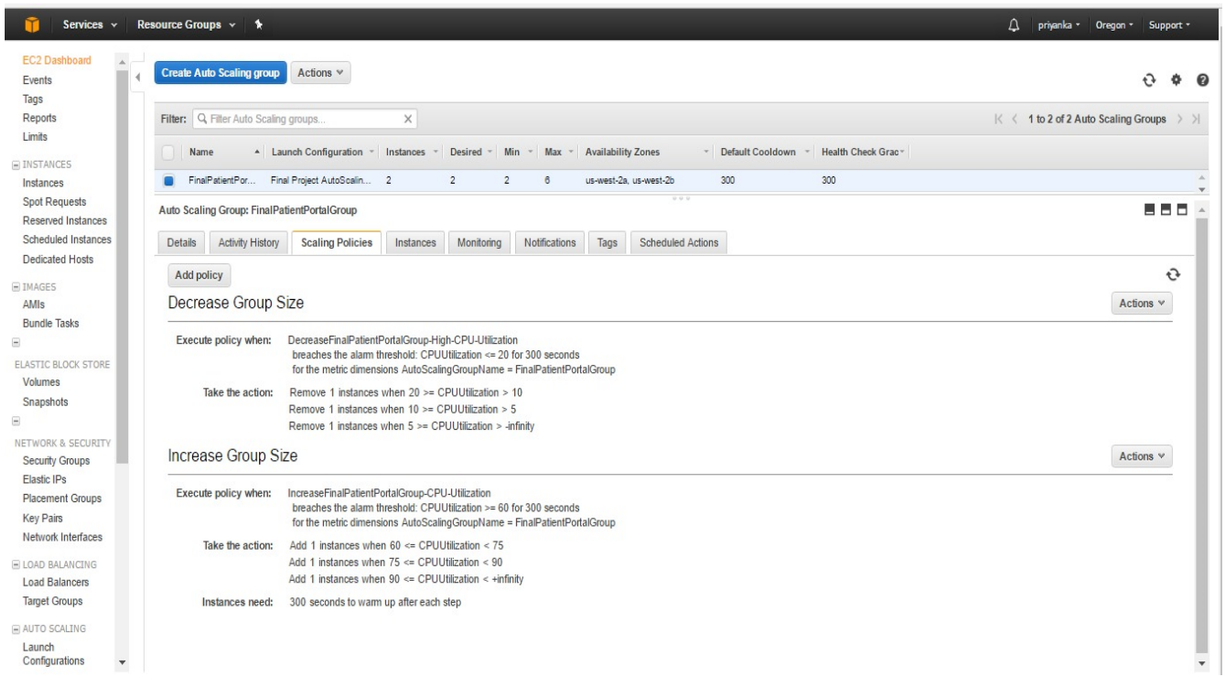
How it works?

- Define an AMI instance and create an Auto Scaling Group to launch instances into.
- Use Cloud Watch to monitor our server(s)/instance(s), and when certain events happen (i.e. events that can trigger an alarm like CPU utilization or DiskWriteOps)
- Launch/Decrease instances based on the AMI template we define while creating the auto scaling group. The EC2 instances launch behind the Elastic Load Balancer (ELB) we define. The ELB will send traffic in a round-robin pattern between all the instances assigned to it, and we can control in real time how many instances we want to launch to cover traffic (high and low). If any of the EC2 instances fails to respond, the ELB will detect it and launch a replacement. When web traffic dies down, instances can be terminated automatically.

Auto Scaling Group Details



Auto Scaling Group – Scaling Policies:



Scaling Policies

Step Scaling Policies:

Step Scaling Policy has been set for scaling the instances based on a group of steps. Below are the policy details.

Step 1:

60% <= CPU Utilization <=75%
The alarm is breached when the threshold of CPU Utilization crosses 60%, a new instance is launched.

Step 2:

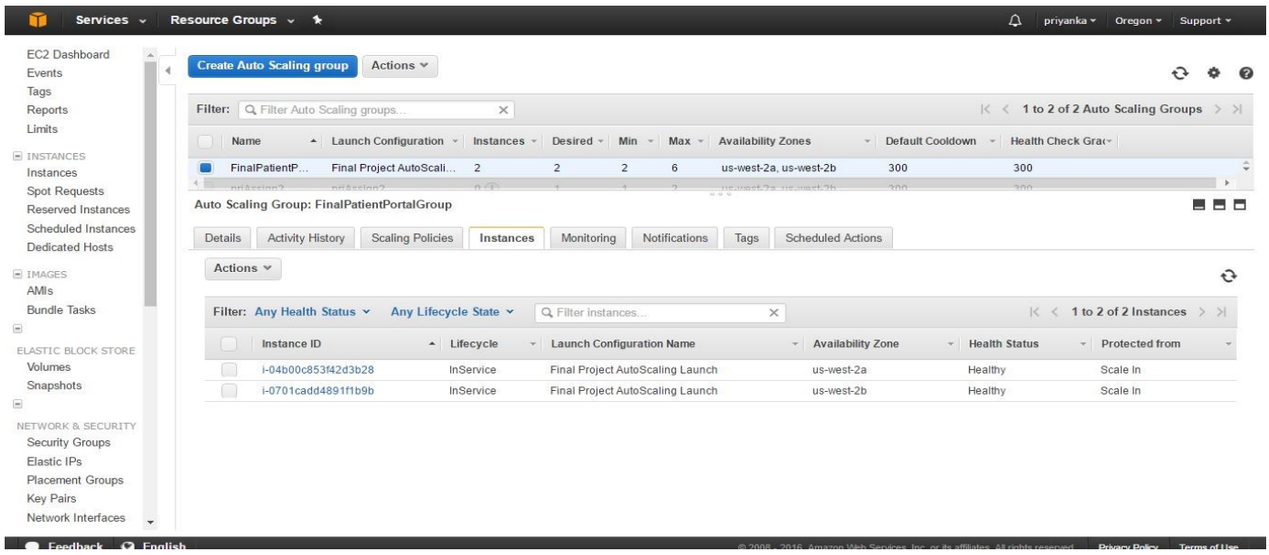
75%<= CPU Utilization <= 90%
The second step adjustment of the scale out policy triggers a new instance when the CPU Utilization crosses 75%.

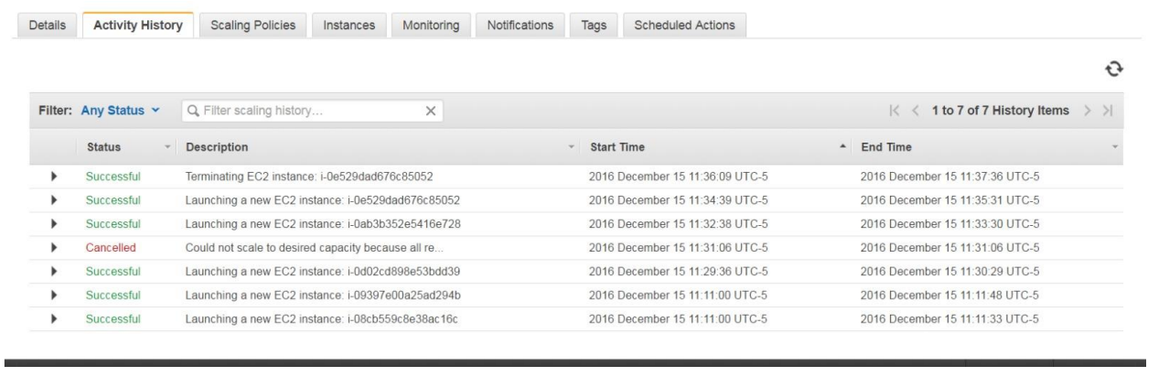
Step 3

90%<=CPU Utilization <=INFINITY

Similarly, scale out takes place when the specified threshold is breached.

Auto Scaling Group – Health/Instances





i. Cloud Watch

Cloud watch enables real time monitoring of AWS resources such as Amazon EC2 instances, Load Balancers and Amazon RDS database instances. Cloud watch collects and processes raw data into real-time metrics.

Cloud watch works with the help of alarms. An alarm can have three possible states.

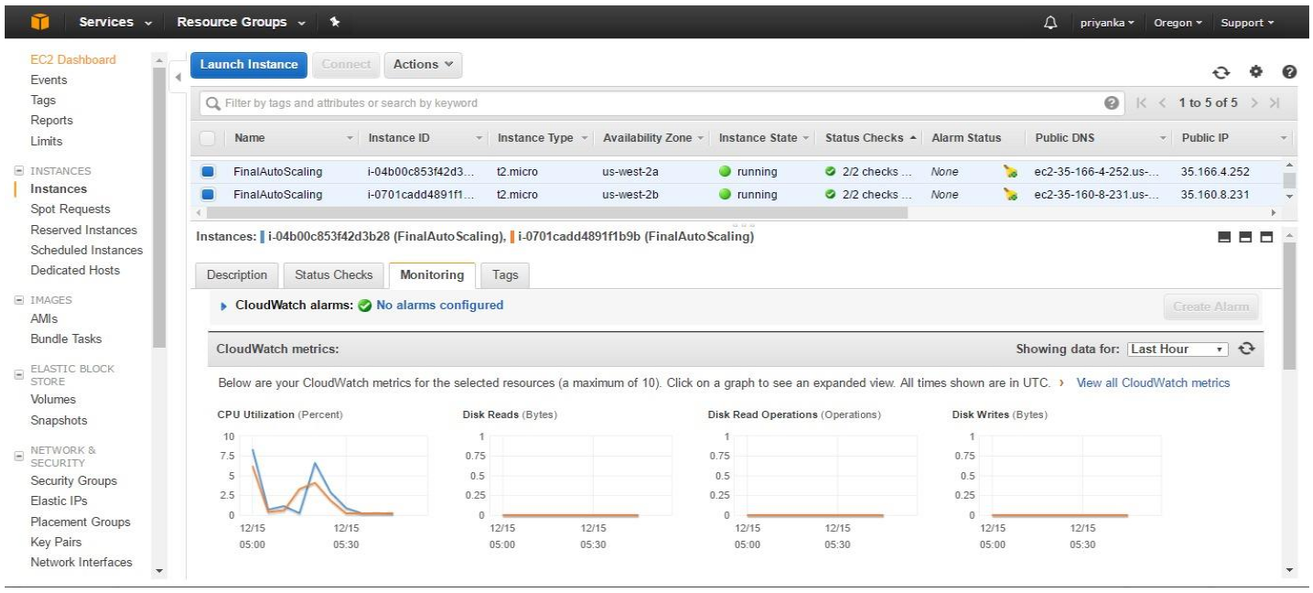
- OK—The metric is within the defined threshold
- ALARM—The metric is outside of the defined threshold
- INSUFFICIENT_DATA—The alarm has just started, the metric is not available, or not enough data is available for the metric to determine the alarm state.

For our application, we have set 2 alarms to trigger. We have chosen the CPU utilization metrics from EC2.

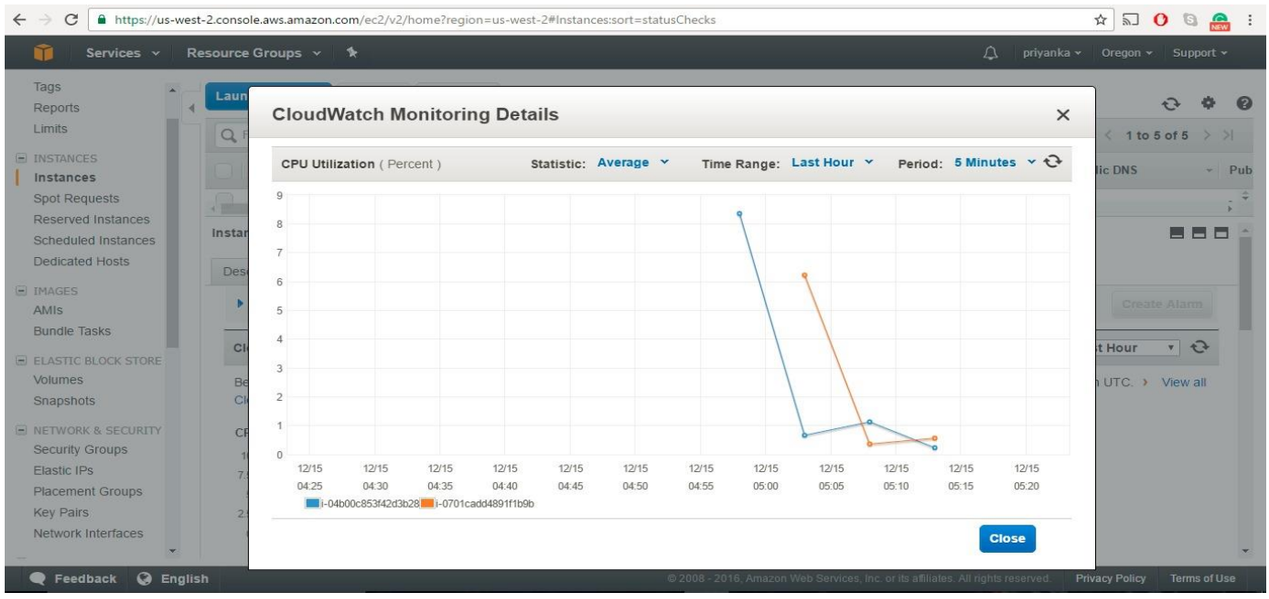
1. If the CPU utilization is > 60% for 5 mins then the auto scaling group will automatically scale up by an instance.

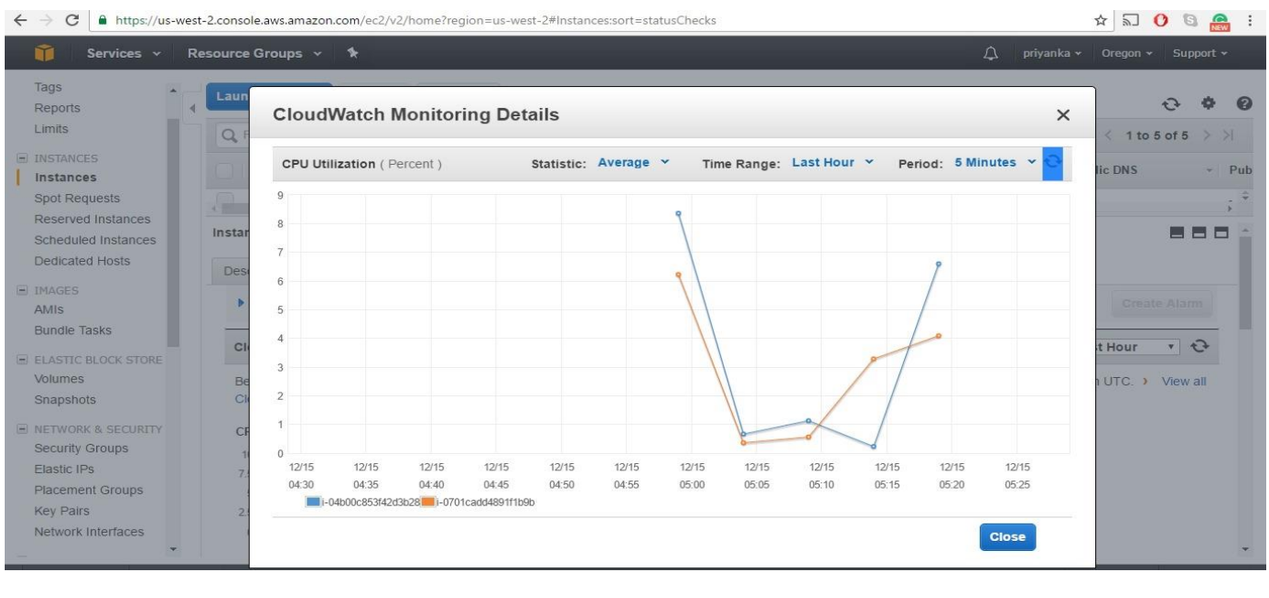
2. If the CPU utilization is <5% for 5 mins then the auto scaling group will automatically scale down by one instance.

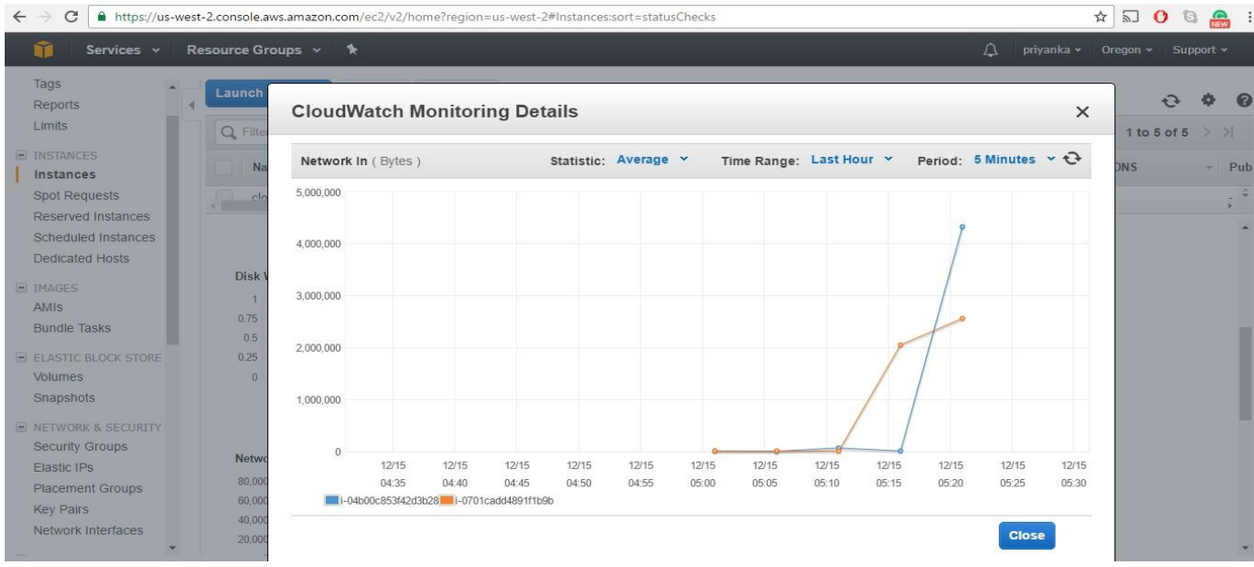
Below is the Cloud Watch Monitoring Tab of the Load Balancers

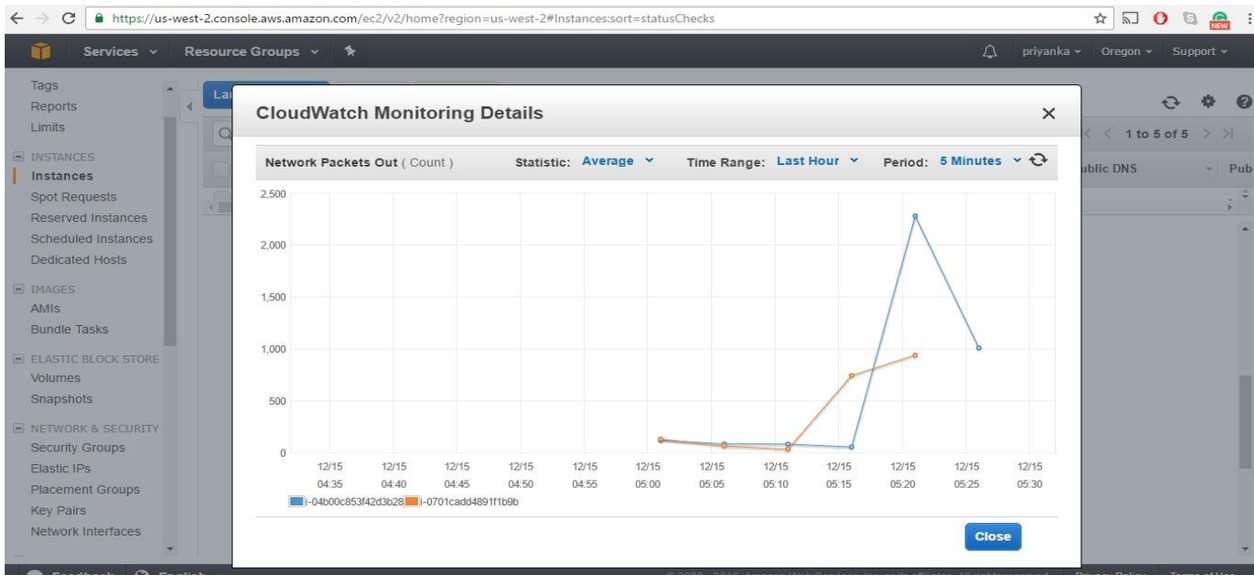


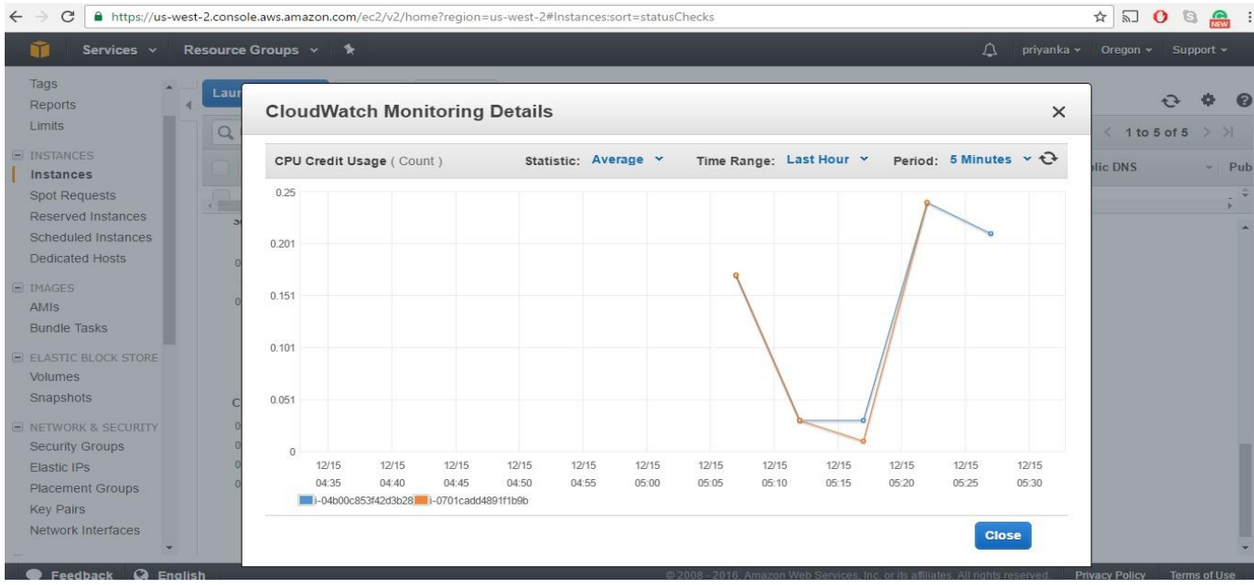
Cloud Watch Monitoring Screenshots



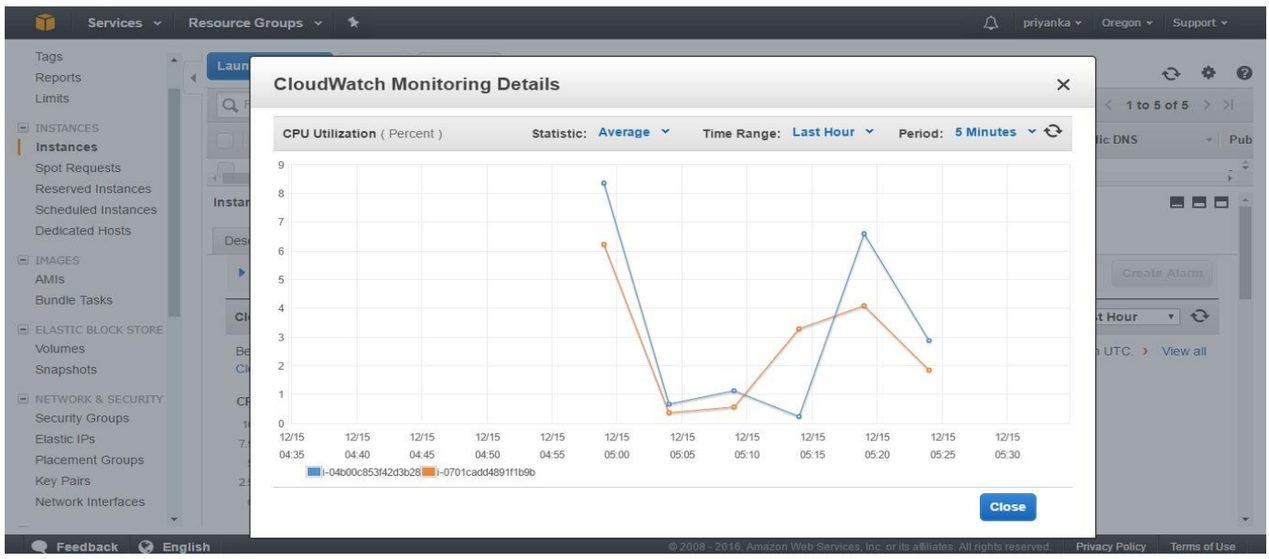



https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:sort=statusChecks
Services
Resource Groups
priyanka
Oregon
Support
Tags
Reports
Limits
INSTANCES
Instances
Spot Requests
Reserved Instances
Scheduled Instances
Dedicated Hosts
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Launch
1 to 5 of 5
CloudWatch Monitoring Details
Network In (Bytes)
Statistic: Average
Time Range: Last Hour
Period: 5 Minutes
5,000,000
4,000,000
3,000,000
2,000,000
1,000,000
0
12/15 04:35
12/15 04:40
12/15 04:45
12/15 04:50
12/15 04:55
12/15 05:00
12/15 05:05
12/15 05:10
12/15 05:15
12/15 05:20
12/15 05:25
12/15 05:30
i-04b00c853f42d3b28
i-0701cadd4891f1b9b
Close


https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:sort=statusChecks
Services
Resource Groups
priyanka
Oregon
Support
Tags
Reports
Limits
INSTANCES
Instances
Spot Requests
Reserved Instances
Scheduled Instances
Dedicated Hosts
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
1 to 5 of 5
CloudWatch Monitoring Details
Network Packets Out (Count)
Statistic: Average
Time Range: Last Hour
Period: 5 Minutes
2,500
2,000
1,500
1,000
500
0
12/15 04:35
12/15 04:40
12/15 04:45
12/15 04:50
12/15 04:55
12/15 05:00
12/15 05:05
12/15 05:10
12/15 05:15
12/15 05:20
12/15 05:25
12/15 05:30
i-04b00c853f42d3b28
i-0701cadd4891f1b9b
Close


https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:sort=statusChecks
Services
Resource Groups
priyanka
Oregon
Support
Tags
Reports
Limits
INSTANCES
Instances
Spot Requests
Reserved Instances
Scheduled Instances
Dedicated Hosts
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
1 to 5 of 5
CloudWatch Monitoring Details
CPU Credit Usage (Count)
Statistic: Average
Time Range: Last Hour
Period: 5 Minutes
0.25
0.201
0.151
0.101
0.051
0
12/15 04:35
12/15 04:40
12/15 04:45
12/15 04:50
12/15 04:55
12/15 05:00
12/15 05:05
12/15 05:10
12/15 05:15
12/15 05:20
12/15 05:25
12/15 05:30
i-04b00c853f42d3b28
i-0701cadd4891f1b9b
Close
Feedback
English
© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

CPU Utilization Final Monitoring



3. Cloud Pricing model for the next 3-6 months

Services | **Estimate of your Monthly Bill ($ 80.41)**

Estimate of Your Monthly Bill

Show First Month's Bill (include all one-time fees, if any)

Below you will see an estimate of your monthly bill. Expand each line item to see cost breakout of each service. To save this bill and input values, click on 'Save and Share' button. To remove the service from the estimate, jump back to the service and clear the specific service's form.

Save and Share

Amazon EC2 Service (US-East)			$ 56.46
Compute:	$ 38.08		
Elastic LBs:	$ 18.30		
Data Processed by Elastic LBs:	$ 0.08		
Amazon RDS Service (US-East)			$ 53.03
Amazon CloudWatch Service (US-East)			$ 0.00
AWS Support (Basic)			$ 0.00
Free Tier Discount:		$	-29.08
Total Monthly Payment:		$	80.41

Individual Components' Pricing

EC2

Services | **Estimate of your Monthly Bill ($ 80.41)**

Choose region: US-East / US Standard (Virginia)

Inbound Data Transfer is Free and Outbound Data Transfer is 1 GB free per region per month

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers. Amazon Elastic Block Store (EBS) provides persistent storage to Amazon EC2 instances.

Clear Form

Compute: Amazon EC2 Instances:

Description	Instances	Usage	Type	Billing Option	Monthly Cost
EC2 instances	2	100 % Utilized/M	Linux on t2.micro	On-Demand (No Cor	$ 19.04
EC2 instances	4	50 % Utilized/M	Linux on t2.micro	On-Demand (No Cor	$ 19.04
Add New Row					

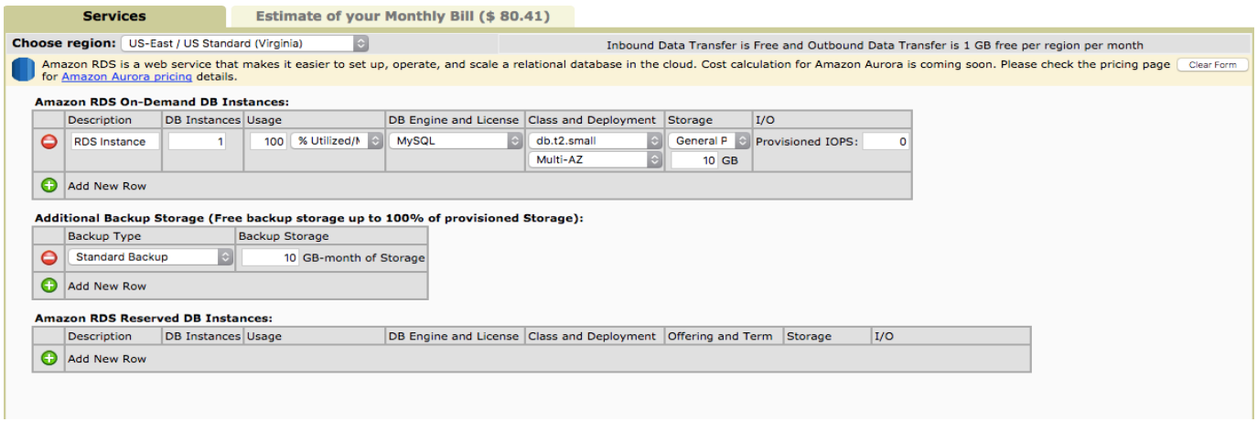
Compute: Amazon EC2 Dedicated Hosts:

Description	Number of Hosts	Usage	Type	Billing Option
Add New Row				

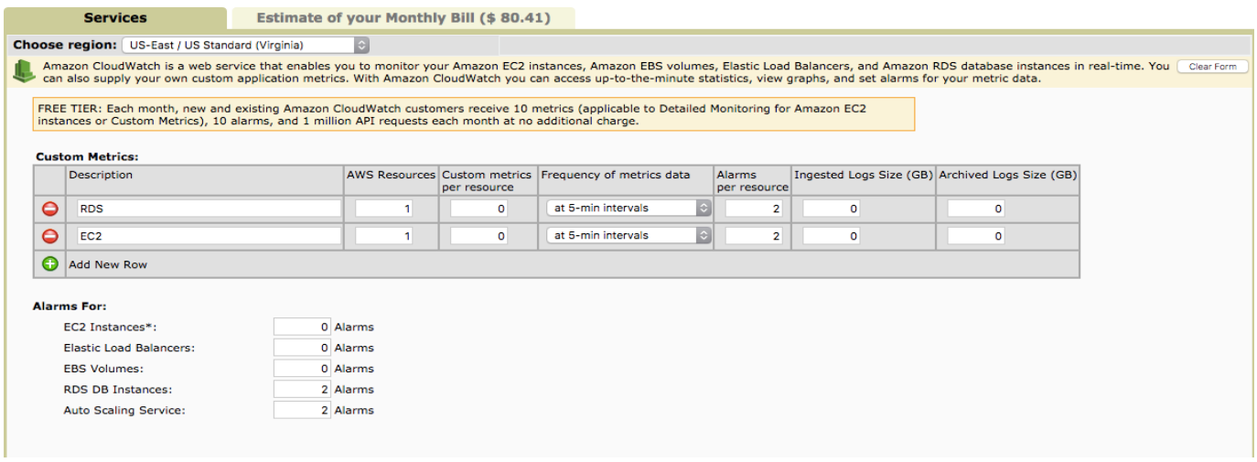
Storage: Amazon EBS Volumes:

Description	Volumes	Volume Type	Storage	IOPS	Baseline Throughput	Snapshot Storage
Add New Row						

RDS



Cloud Watch



5. Disaster Recovery Procedures

a. Pilot Light Model for Quick Recovery

The term pilot light is often used to describe a DR scenario in which a minimal version of an environment is always running on the cloud. Our infrastructure has been built on Amazon RDS which allows for having a master and redundant read replica which is on standby. To provision the remainder of the infrastructure to restore business-critical services, we have preconfigured servers bundled as AMI's (Amazon Machine Images), which are ready to be started up at a moment's notice. When starting recovery, instances from these AMIs come up quickly with their pre-defined role (for example, Web Server or App Server) within the deployment around the pilot light model. We are using ELB (Elastic Load Balancing) to distribute traffic to multiple instances. Our clients interact with the application through the load balancer, so that any instances going down or coming up wouldn't have any impact for them.

b. Preparation Phase

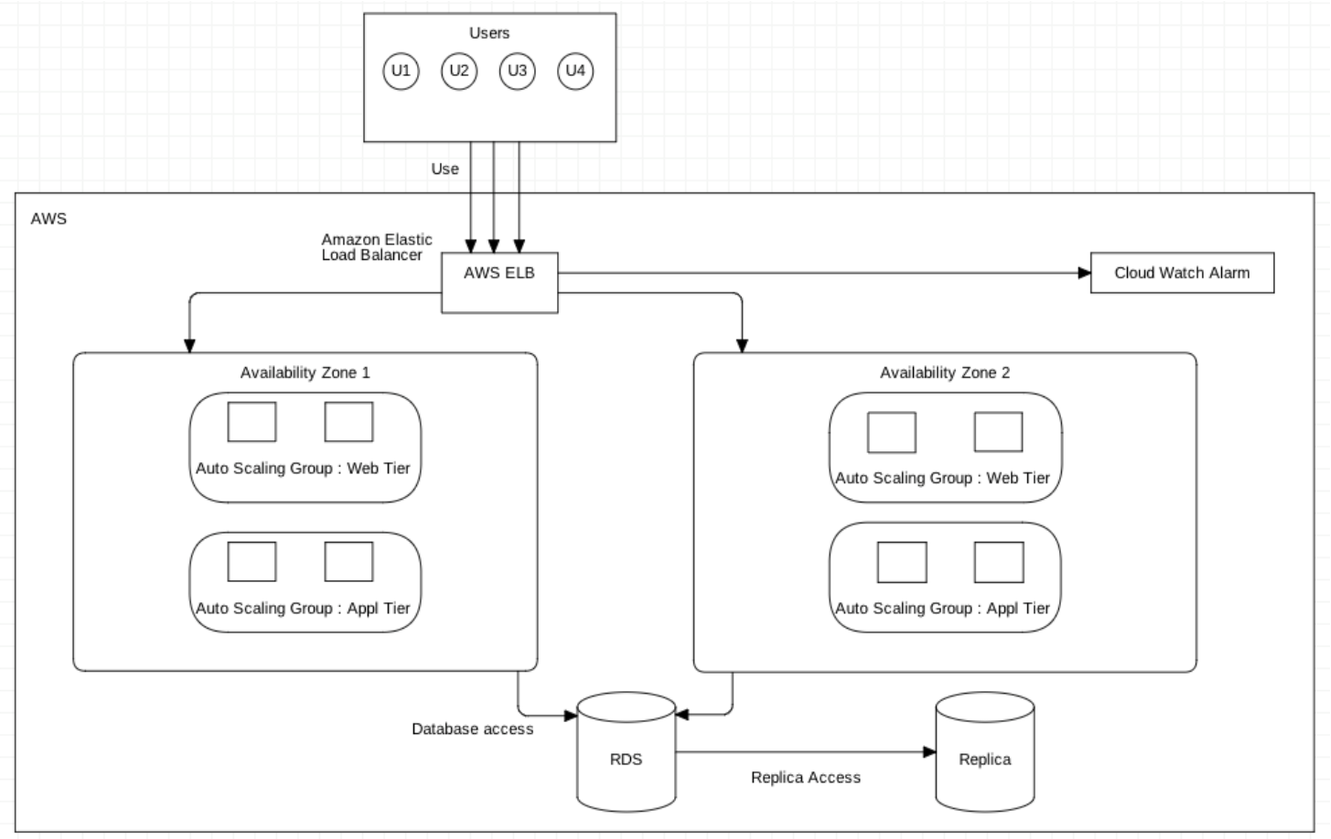
We are using RDS to have a read replica of the Database, since the Database is small we have the replica of the entire Database. Any changes to the operating system, application and scripts are periodically updated and stored as an AMI, all our instances are launched from this AMI instance. We monitor the health of these instances through Cloud Watch to check their status

c. Recovery Phase

To recover the remainder of the environment around the pilot light model, we start our systems from the AMIs within minutes on the appropriate instance types. We also make sure that we have at least two instances running at any time and both of them on different subnets. After recovery we again ensure that the redundancy is restored in RDS

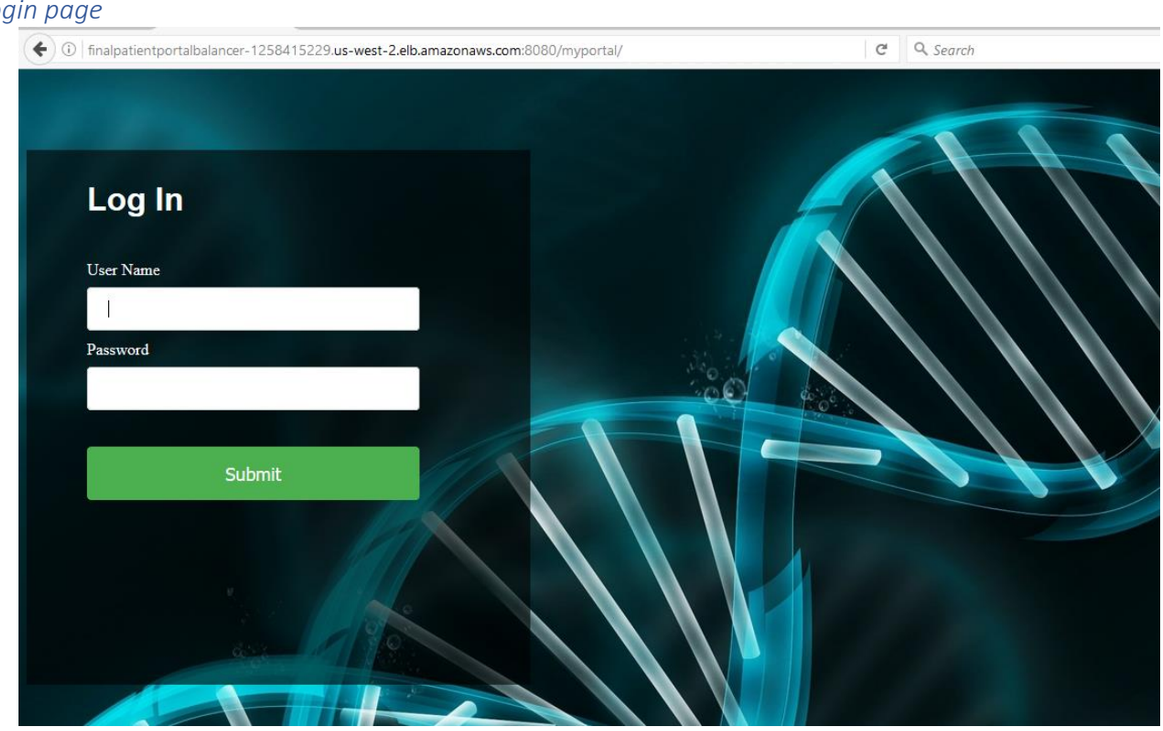
Also we use auto scaling policies to handle traffic by launching more instances and scaling out our architecture

Visio Diagram of the Architecture

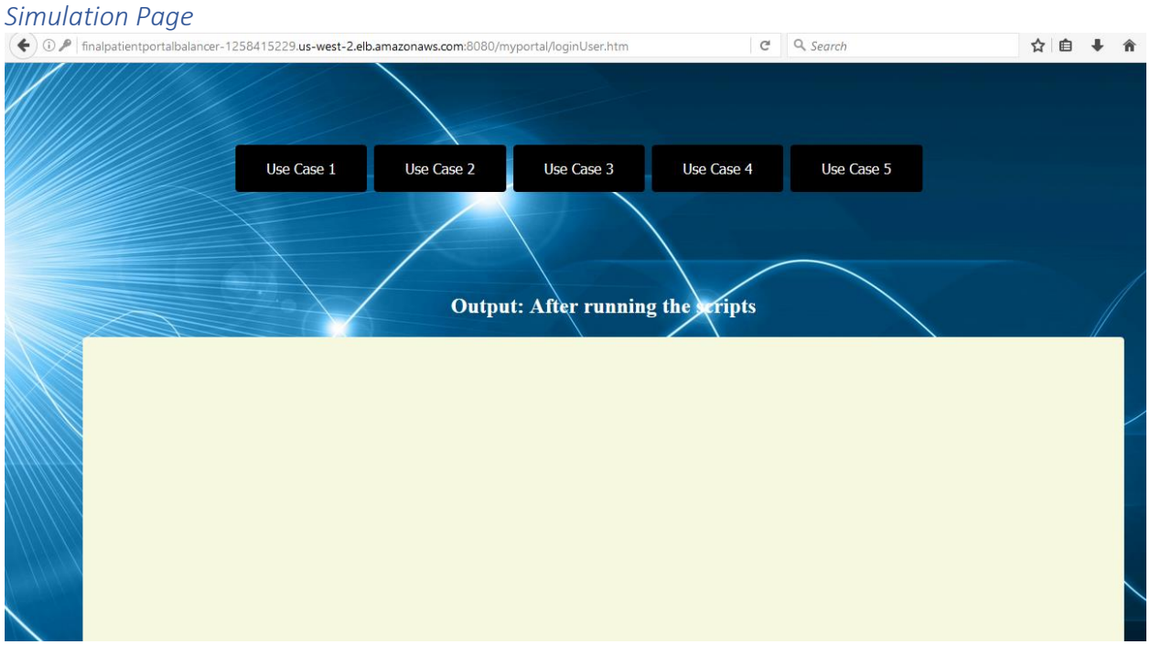


6. Application Screenshots

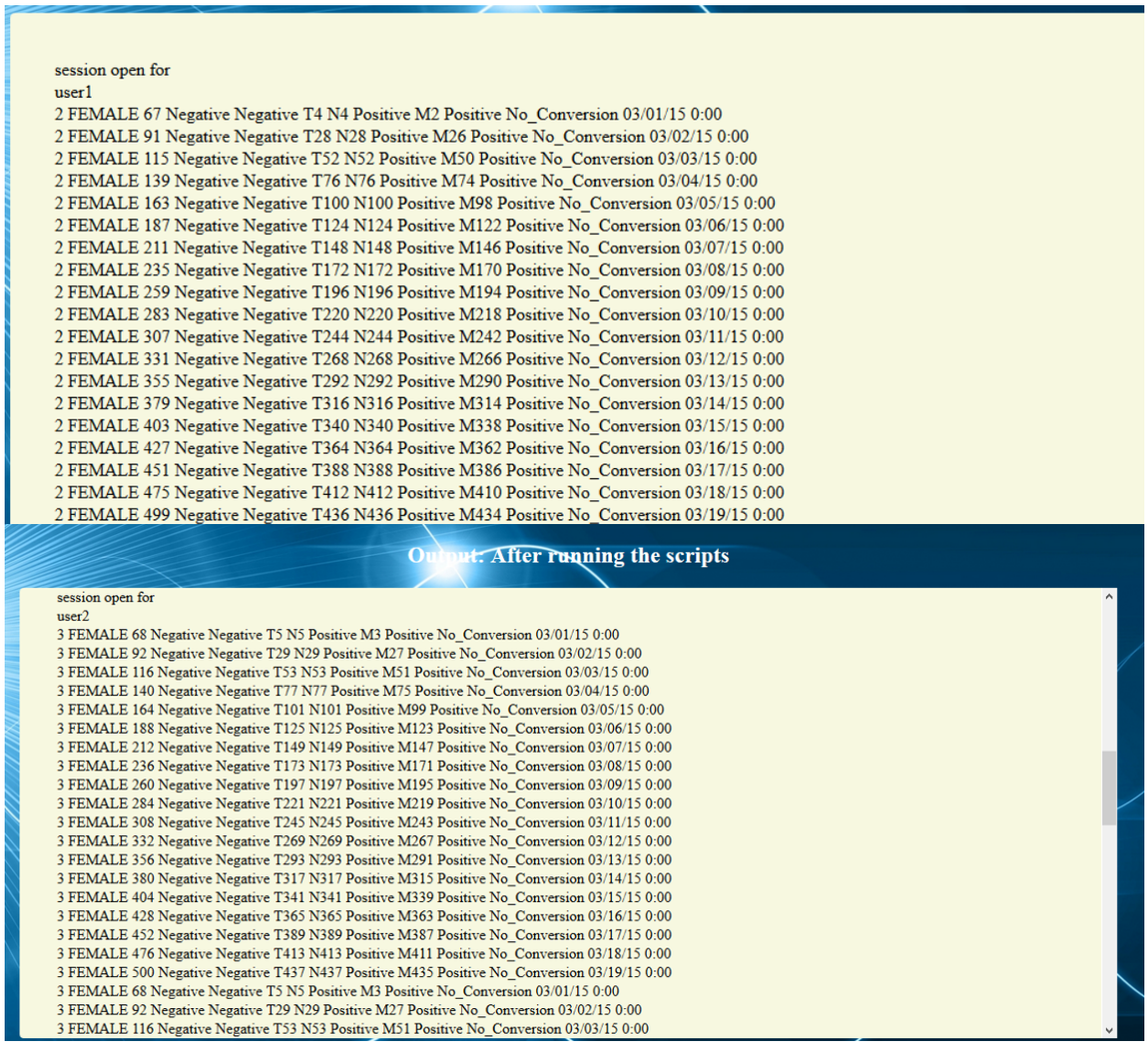
Login page

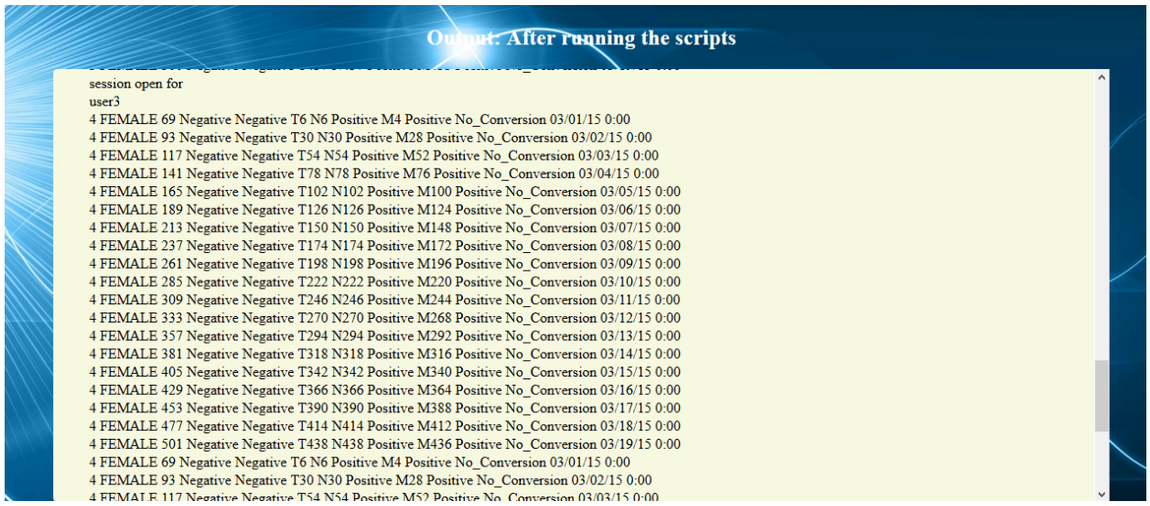


Simulation Page



Use Case Execution





7. References:

https://aws.amazon.com/documentation/
https://en.wikipedia.org/wiki/