CAB432 Assignment 2 Individual Report

Name: Yanmei Zeng
SN: 10307389
Partner: Yonathan Cahyadi (10149953)

1. Statelessness, Persistence and Scaling (See figure 1)

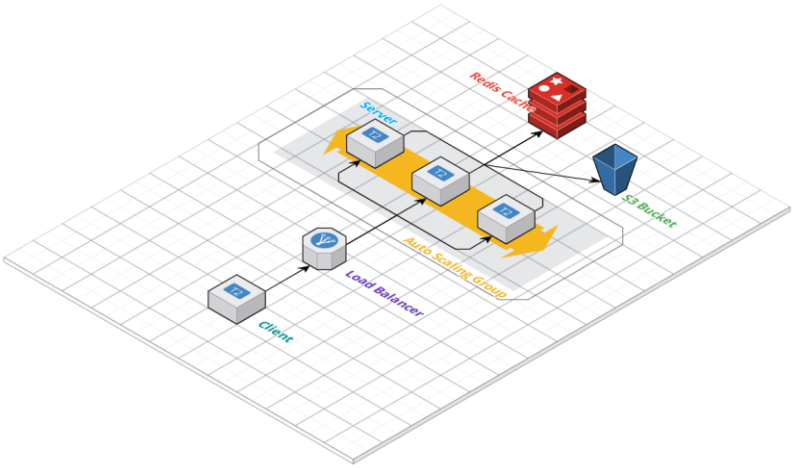

Figure 1

1.1 The mechanisms for handling application state and their robustness when instances fail.

For handing state, the app using Redis Cache for short term storage, with Replica and Backup 3 Node can prevent Primary Node fail. For long term storage the app using AWS S3 bucket, it can prevent Redis Cache storing fail.

1.2 compromises from true statelessness and the reasons for these choices

With the client side, user must need authorization token, it keeps user login when the page refresh, and promise to store processed data in Redis and S3 bucket. In addition, can remove token to safely logout

1.3 The choices made with respect to persistence and their suitability for this application

The app is using Redis for a short term and AWS S3 bucket for a longer-term storage. The reason why we are store processed data this way is if the data is too large instead of taking a long time for server t response, it increases the efficiency. Another reason is its secure data storing in case there is a failure.

1.4 Possible better alternatives feasible in the university context.

We can use MongoDB database for adding up data storing security and optimize the response time.

1.5 Approach to application scaling, and the suitability of the metrics and the thresholds used

Based on CUP utilization, Network In and Network Out to Scaling this app, in addition, the server would scale if CPU utilization is over 50% or Network In and Network Out is more than 10MB. As this

app need to hand lot of data processing for getting lyrics and calculate the sentiment rate, its mandatory to scaling the app.

2. The Global Application (See figure 2)

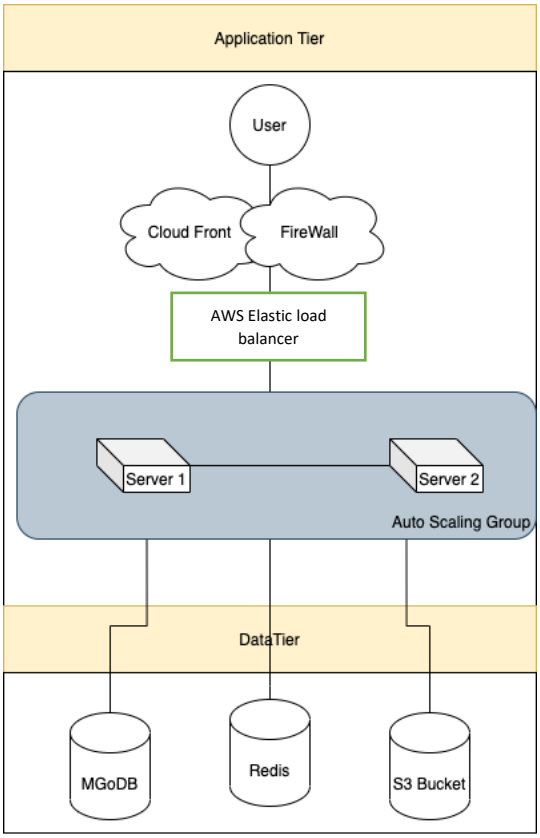

Figure 2

For make the app global user base, we need to consider mainly 2 aspects. First, handle the incoming network traffic and huge data processing requests. Second, security issues. There are services can be used to handle these 2 main issues. AWS Elastic Load Balancing can automatically distribute incoming traffic across multiple targets – Amazon EC2 instances, containers, IP addresses, and Lambda functions – in multiple Availability Zones and ensures only healthy targets receive traffic. Elastic Load Balancing can also load balance across a Region, routing traffic to healthy targets in different Availability Zones. In addition, it also adds up security as it works with Amazon Virtual Private Cloud (VPC) to provide robust security features, including integrated certificate management, user-authentication, and SSL/TLS decryption. Together, they can give the app flexibility to centrally manage TLS settings and offload CPU intensive workloads from the applications. For security issues can be solved by configuring a logging service encryption and firewall.