CSE - 546 - Project 1 - Portfolio Report

Priyadarshini Ramakrishnan(ASU ID - 1225407339)

My primary responsibilities in this project are centered around implementing the app tier functionality and configuring the Flask web application, which was developed by one of our team members, on the EC2 server to make it accessible via the internet. To accomplish these two tasks, firstly, I created a Python script called "app_tier.py" that utilizes the "boto3" Python library, an AWS SDK, to interact programmatically with SQS queues and S3 buckets. The script receives raw Byte data of the image from the request SQS queue, reads and deletes it using the "sqs_receive()" function, and then converts it to image data using the "convert_byte_to_image()" function. The resulting image is then stored as an object in the S3 bucket using the "s3_upload()" function, with the key of the object being the file name and the object itself being an image file.

After the image is processed using the image classification model provided to us, the top-1 result of the classification is sent to the response SQS queue using the "sqs_send()" function for consumption by the web tier. The classification results are also stored as objects in the output S3 bucket using the "s3_put()" function, with appropriate tags containing the result values being added to these objects.

To ensure that the app tier functionality could be utilized by our autoscaling groups during dynamic scaling, I deployed the app_tier script on an AWS EC2 instance and created an AMI of the instance. Additionally, I created a system service that launches the app_tier Python scripts every time a new instance is launched, or an existing instance is rebooted.

Furthermore, I hosted the web application on EC2 using Nginx as a reverse proxy and Gunicorn as a request gateway service. Initially, I created a virtual Python environment and installed the necessary Python libraries (namely, Gunicorn, Gevent, Flask, and Boto3) in the active Python virtual environment. I then installed Nginx on the EC2 web instance to act as a reverse proxy server to route the traffic to the web gateway service Gunicorn. Next, I created a system service called "web-tier.service" to run the Gunicorn gateway during the boot of the EC2 instance. I also modified the default Nginx configuration to enable port forwarding for HTTP (port 80), GET, and POST requests. The above services were started and enabled to automatically start during system reboot or server launch. Lastly, the inbound rules of the webserver EC2 instances were set up to allow HTTP requests, which made the web application accessible via the internet.