Cloud Computing : CSE 546

Project 1
Shatrughn Gupta (1215160518)
Shireen Abraham (1215126536)
Ripudaman Teja (1214899621)

The details about the application are as follows:

- Programming language : Java
- Application framework: Spring
- Package handler: Maven

Steps to run the code:

1. Any constants like queue url, message visibility timeout, regions, etc can be changed in the file Constants.java.
2. Build a jar from the Web - Tier (project1) code. This can be done either manually or through maven by running “Maven clean” and then “Maven install”.
3. If running locally, type “java -jar ‘jar-name’ ”.
4. If running on EC2, copy the jar to the EC2 instance and run above command.

Design:

Web Tier (project1):

1. **App.java** - main class is contained in this file.
2. **AppController.java** – handles all the client requests . It primarily serves 3 functions:
    a. Generating and assigning a unique id to the request.
    b. Delegation of request to the sqs handler, which puts the request in the input queue.
    c. Invoking the response handler, which polls the output queue continuously for a corresponding response. Once received, sends it back to the client. To avoid continuous polling, we have given the thread, a pause of 3 seconds.

3. **ResponseProcessorImpl.java:** polls the output queue for the response of a request and serves it back to the **AppController.java**

4. **AutoScalerServiceImpl.java:**
    a. Scales the application based on number of requests in the queue. If the requests are more than current working application instances, starts a new instance.
5. **Constants.java:** contains all the static information like urls for the queues, bucket url, etc.
6. **Ec2OperationsImpl.java:** serves the following functions:
    a. Copying the configurations from a static image to new application instances and starts them.
    b. Getting a list of available application instances.
    c. Assigning names to the instances.
7. **SqsOperationsImpl.java:** provides implementation to the SqsOperations interface. It serves the following responsibilities:
    a. Adding the request to the input queue.
    b. Reading the messages from the output queue.
    c. Deleting the messages from the queue.
8. **AwsConfig.java:** is a configuration file which builds and provides objects to use amazon’s web services api.
9. **Pom.xml**: contains information about the project and configuration details used by Maven to build the project.

Application - Tier (worker):

1. **WorkerApplication.java:** main class for the project. It starts the application tier and processes the requests.
2. **S3OperationsImpl.java:** provides an implementation to the S3Operations contract. It serves the following responsibilities:
    a. Creating the buckets if it does not exist.
    b. Putting the response in the bucket.
3. **SqsOperationsImpl.java:** provides an implementation to the SqsOperations contract. It serves the following responsibilities:
    a. Reading the request from the input queue.
    b. Deleting the request from the input queue.
    c. Adding the response to the output queue.
4. **VideoServiceImpl.java:** provides an implementation to the VideoService contract. It downloads the video from the given url, processes it, performs object detection and stores the result in the S3 as well as the output queue.