CS 553 PROGRAMMING ASSIGNMENT-1

EVALUATION DOCUMENT

ROHIT SINGH

A20361198

1) Install java

Install java jdk on the instance if it doesn't exist.

sudo apt-get update
sudo apt-get install openjdk-7-jre

java -version
nano ~/.bash_profile

export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-amd64;
export JRE_HOME=/usr/lib/jvm/java-1.7.0-openjdk-amd64;
export PATH=/usr/sbin:/usr/bin:/sbin:/bin:$JRE_HOME/bin;

---

2) Copy the Credentials

To run the experiments you need to upload the .aws folder to the instance. One can use the following scp command to copy the folder to the instances.

scp –r –i <key.pem> ~/.aws username@public_dns:~/

Eg:
scp -r -i "hadoop.pem" ~/.aws ubuntu@ec2-52-36-82-52.us-west-2.compute.amazonaws.com:~/

---

3) Create Sleep File

To create sleep tasks use the following script

./create_sleep_tasks <FileName> <Nos of Instructions> <Time per Instruction>

E.g.:

./create_sleep_tasks "sleep$j$i.txt" $val $j

The command line variables used in the script are

- $1 file name
- $2 instructions
- $3 time

---

4) Run the Local Experiment:

To execute this experiment, the following script can be used:

chmod 700 implement
./implement

However, to only run the jar file the following syntax can be used provided the file has been created.

java -jar LOCAL.jar <#Threads> <FileName>

Inside LOCAL.jar:

Here, the local code emulates the client and the threads as the workers when compared to the remote experiment.

LocalClient.java: Manages the Queue (create, push and pop) and deploys jobs to the Threads.
LocalThread.java: This program runs the sleep command for the threads by implementing the run().

---

5) Run the Remote Experiment:

To execute this experiment, first the client needs to be executed so that the Queue can be created with the tasks:

java -jar CLIENT.jar <FileName> <QueueName>

Inside CLIENT.jar:

Here, the client code reads the tasks from the file and enters them into the SQS, then enters the SQS url into the database.

ClientAmazonDynamoDB.java: This file does all the connections with the DynamoDB tool and reads the file. It calls the ClientSimpleQueueService.java to do the queue operations.
ClientSimpleQueueService.java: This file does all the connections with the SQS tool.

To execute the workers so that the workers can start doing the tasks present in the Queue:

java -jar WORKER.jar

Inside WORKER.jar:

Here, the workers read the url of the queue from the database, and then starts reading from the queue. After it has read an item it check from the database if the item is a duplicate, and then does the task. After, the tasks have been done it replies back to the response queue.

AmazonDynamoDBSample.java: Manages the database deploys jobs to the Workers.

SimpleQueueServiceSample.java: This file does all the connections with the SQS tool.

WorkerThreadExecuter.java: This file does the pull from the SQS using the file SimpleQueueServiceSample.java, and checks if the item is duplicate or not using the file AmazonDynamoDBSample.java.

LocalThread.java: This program runs the sleep command for the threads by implementing the run(), in this case the thread count is 1.

---

6) Animoto Clone

Install the ffmpeg on the system. The commands to do so are as follows:

- remove the current ffmpeg by running:
  sudo apt-get remove --purge ffmpeg
- Add ppa trusty-media:
  sudo apt-add-repository ppa:mc3man/trusty-media
- Update repo:
  sudo apt-get update
- Finally install it:
  sudo apt-get install ffmpeg

To execute this experiment, the following command can be used:

java -jar Animoto.jar

Inside Animoto.jar:

SaveImage.java: downloads the images by reading the urls from the file. Calls the AnimotoS3Transfer.java to save the video and animotoSimpleQueueService.java to push the S3 url.
