Assignment 1 README

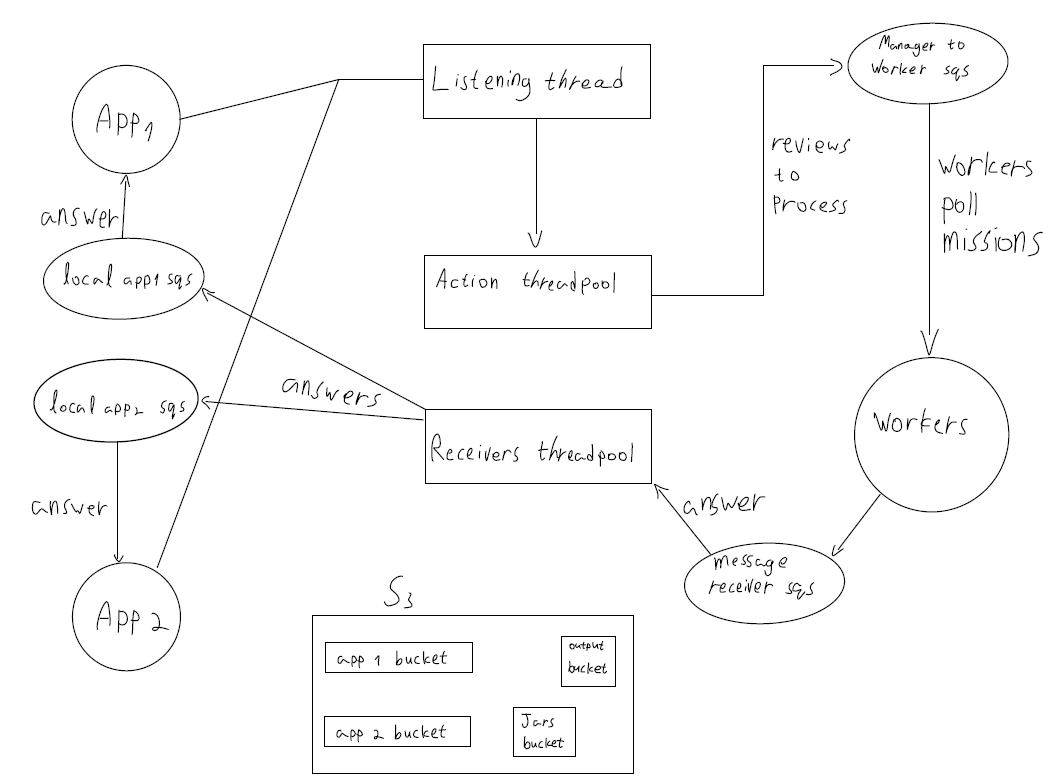
Shachar Sultan 315780114, Yovel Sazgar 208716548.

**Program flow:**

1. Some local application starts an ec2 instance that represents the manager if it isn't already running. Then, it uploads a text file to a bucket and sends a message through the global sqs.
2. The manager composed of 3 threads:
   * Listening Thread: waiting for the global sqs to get messages and receive messages from the local apps.

It transfers every message to the QueueOfRequets, there the action thread gets the messages and starts processing them.

* + Action Thread pool: Each thread polls a message from the QueueOfRequets, takes the file that local application uploaded earlier, separates it to reviews, and then calculate how much workers need to start, according to the active workers, and sends them their mission.
  + Receiving thread pool: The threads receiving answers from the workers by the "Message Receiver SQS" and upload the final answer to the s3 answer bucket. Then, they send a message to the local application that the job is done, and the answer is in S3.

1. The workers get reviews from the "Worker To Manager SQS". They perform the Stanford nlp algorithms on the review they got and send back the answer to the message receiver sqs and then it gets to the Manager.
2. The manager sends the final answer to the local application, and then the application downloads the final answer, processes it and creates the final output file.
3. After the workers finish processing all the tasks, the manager sends them a terminate message and then terminates itself.

**Dependencies:**

Ami: ami-00e95a9222311e8ed type: large

**Running times:**

1 local application that sends 1 input file to the manager finishes in about 6 minutes. 2 local applications: the first one sends 1 file, the second one 2 files, finishes in about 18 minutes. n is 100.

**Scalability:**

Every local application has one queue that receives answers from the manager and uploads to the manager.

The manager has a queue that receives missions from the local application and distributes them to action threads. It also has a hash map that keeps the local apps' sqs url as a key and the number of reviews to process as a value. Another hash map keeps the local apps' url as a key and the answer file name for the local app as a value.  
In addition, the manager has 2 queues to send messages to the workers an receive messages from the workers.

**Persistence:**

When the manager sends a mission to the workers, it sends it with a visibility timeout of 10 seconds. Which means that when a certain worker takes this mission no other worker can process it as well at the same time, to prevent duplicates. So, if a worker dies in the middle of a mission. After 10 seconds another worker can take this message and process it. The worker deletes the message only after it finishes processing it.

**Threads:**

We are using one thread to process more than one message at once, this is the action thread.

We are also using the receiving thread for that purpose, but here we use it for the answers.

The listening thread is used for getting messages from the local applications.

**Workers:**

Some of the workers work harder than others and process more reviews than others. This is because some reviews are shorter than others, so it takes the Stanford nlp algorithm more time to finish running.

**Running the program:**

To run the program, you need to add those arguments to the command line arguments:

* The input files that you want to process, for instance "input1.txt input2.txt".
* The name of the output file.
* Number of tasks per worker(100)
* If you want to terminate, add the word "terminate".