Mitchell Kane

2/29/20

CSS 436 Cloud Computing

Program 4

Program 4 serves to integrate multiple facets of cloud computing into one assignment. In this program I have integrated AWS S3 storage for storing a datafile, AWS DynamoDB for hosting a no sql database, and AWS Elasticbeanstalk for web app hosting. I chose to create this program in python with the boto3 extension, as utilizing C# with Visual Studio is less pleasant to do on my Mac. A high-level observation of the program describes it as this: a URL endpoint containing data is reached, the data is extracted and copied in an AWS S3 bucket associated with the Elasticbeanstalk environment, while at the same time the data is parsed, cleaned, and entered into a no sql AWS DynamoDb database for future queries. To the user, a basic website is presented offering options to load data, query the data, and delete the data. All possible relationships of which buttons are selected in which order have been considered and provide appropriate feedback on the website.

Specifically, the webpage presented to the user offers, from top to bottom, two text boxes labeled “Last Name” and “First Name” with a “Submit” button to submit a query, followed by a “Load” button and finally a “Delete” button. When the “Load” button is clicked, the program will retrieve the data from the endpoint URL, copy it to an S3 storage bucket, and enter into the DynamoDB. If it is pressed again, checks will be made along the same path to ensure data is updated if necessary. The process of loading includes multiple steps, so it could very possibly take a minute or more to complete. Once it is complete, queries are able to be made against the data in the database. As per the specified requirements of the assignment, a user may query using a Last Name, a First Name, or both. Additionally, the user my also query by the beginning portion of a First Name. For example, if the true First Name is William, a query for the First Name item “Will,” “Wil,” and “Wi” are acceptable. For all query entries, both upper and lower case submissions are acceptable.

If during test, if the user were to modify the data at the endpoint, this should not disrupt the service. This is because the retrieval of the endpoint data is made regardless, followed by checks and or updates to ensure current data is available to query against. When the Delete button is selected, the S3 storage bucket and the DynamoDb are all removed. Once again, at any point, a press of any button on the webpage should provide useful operational feedback.

Service Level Agreement (SLA):

EC2 instances promise a SLA of 99.9% availability, with the identification of credits (refunds) if that metric is not upheld (<https://aws.amazon.com/compute/sla/>), DynamoDb promises 99.99% availability with similar refund possibilities (<https://aws.amazon.com/dynamodb/sla/>), and S3 promises 99.9% availability with refunds (<https://aws.amazon.com/s3/sla/>).

Seeing as this program utilizes all 3 services, the overall SLA is equal to the product of these individual availabilities: 99.9 \* 99.99 \* 99.9 = 99.79% availability.

At the moment of this program, the service is a “toy.” If it were to scale to a production level, the resources required could be acquired one of two ways, by scaling up or scaling out. Scaling up involves increasing the computational capacity of the machines. Scaling out involves increasing the number of computational machines to handle the increased load. Being that this program is of a very low computational load, scaling up would not be of much benefit. Therefore, scaling out is how the increased traffic would best be handled.

Monitoring of resources:

The elastic beanstalk (EB) is monitored by specialized tools provided in the EB console. Namely, these tools monitor environment health, CPU utilization, network traffic in (in bytes) and network traffic out (in bytes). The DynamoDB provides tools that monitor the amounts of database reads and writes, shown as a percentage of the capacity allowed for each. S3 buckets are monitored or measured by a few select metrics, such as the daily storage capacity (measured in bytes), the number/traffic of requests for objects in buckets and the number and size of objects that are pending replication as well as how long that replication takes. These S3 metrics allow the owner to identify and act on operational issues.

Assumptions:

No language installations should be required for operation of this program. The only configuration required is that the user has proper AWS user credentials on their machine and the AWSCLI installed.

The website URL endpoint for this program in us-west-2 is:

<http://program4appdevmtk61-env.eba-ycpcmekf.us-west-2.elasticbeanstalk.com>

The name of the public S3 bucket, for confirmation of file insertion and deletion is:

elasticbeanstalk-us-west-2-773350788653

The DynamoDB table name in us-west-2 is:

0727916mtkuweduDataMemberscss436

When the “Load” button is clicked:

WAIT for the message to the that “Loaded data to s3 bucket, then dynamodb” or queries and “Delete” will not work appropriately.

Similarly, when “Delete” is clicked:

WAIT about 30 seconds for AWS resources to actually reflect the changes that took place.

If you are having difficult, “Delete” all data, wait one minute, and start over again.

Program design diagrams:

Loading data into storage objects

A picture containing text

Description automatically generated

Submit Query on DynamoDB

A close up of a piece of paper

Description automatically generated

Deleting all data

A close up of a piece of paper

Description automatically generated