Contents

[Site Location and How to Use 2](#_Toc530567170)

[Locations 2](#_Toc530567171)

[Underlying code and files 2](#_Toc530567172)

[application.py 2](#_Toc530567173)

[Clear.py 2](#_Toc530567174)

[Load.py 2](#_Toc530567175)

[Query.py 2](#_Toc530567176)

[requirements.txt 2](#_Toc530567177)

[index.html 3](#_Toc530567178)

[index.css 3](#_Toc530567179)

[Design 3](#_Toc530567180)

[How will it scale with load? 4](#_Toc530567181)

[How do you monitor? 4](#_Toc530567182)

[Estimate of your SLA 5](#_Toc530567183)

# Site Location and How to Use

* <http://application4-dev.us-west-2.elasticbeanstalk.com/>

To use, click the Load Data to pull the input.txt file from the AWS CSS490 bucket. It will then be stored on my S3 bucket and parsed into DynamoDB. Clicking this button multiple times will reload the file and append the items to the current DynamoDB.

Clicking Clear Data will delete the input.txt file on my S3 bucket and clear the DynamoDB table.

Lastly, you can query the data on the DyanmoDB by entering a first, last, or both names into the text boxes and clicking Query. This will take you to a new page with (or without data) with a back button under the data found.

# Locations

* + S3 Storage
    - <https://s3.console.aws.amazon.com/s3/buckets/css490assignment12/?region=us-west-2&tab=overview>
  + Dynamo DB
    - <https://us-west-2.console.aws.amazon.com/dynamodb/home?region=us-west-2#tables:selected=Program4Table;tab=overview>

# Underlying code and files

## application.py

This file is the main driver of the program. It uses the Flask framework to dynamically load HTML files and associated with them. In addition, this file tells a client what file/method to use when going to different links.

## Clear.py

This file contains the methods related to clearing the table and S3 storage. The clearTable method will empty the DynamoDB table and leave the existing primary key. deleteS3File will remove the input.txt file from my S3 bucket.

## Load.py

This file contains the method related to loading a file from the CSS490 bucket and storing it into my bucket, as well, it can parse the file into the DynamoDB. copyFileFromS3 is the method responsible for moving the input.txt file between buckets. parseFileToDB will parse the file in my S3 bucket and store the elements in DynamoDB

## Query.py

This file contains the method to query DynamoDB. The method is called getData. It will return a HTML table of the results found or return a h1 tagged notification that no data was found.

## requirements.txt

This file is used by Elastic beanstalk to know which python dependencies are needed when setting up the virtual environment.

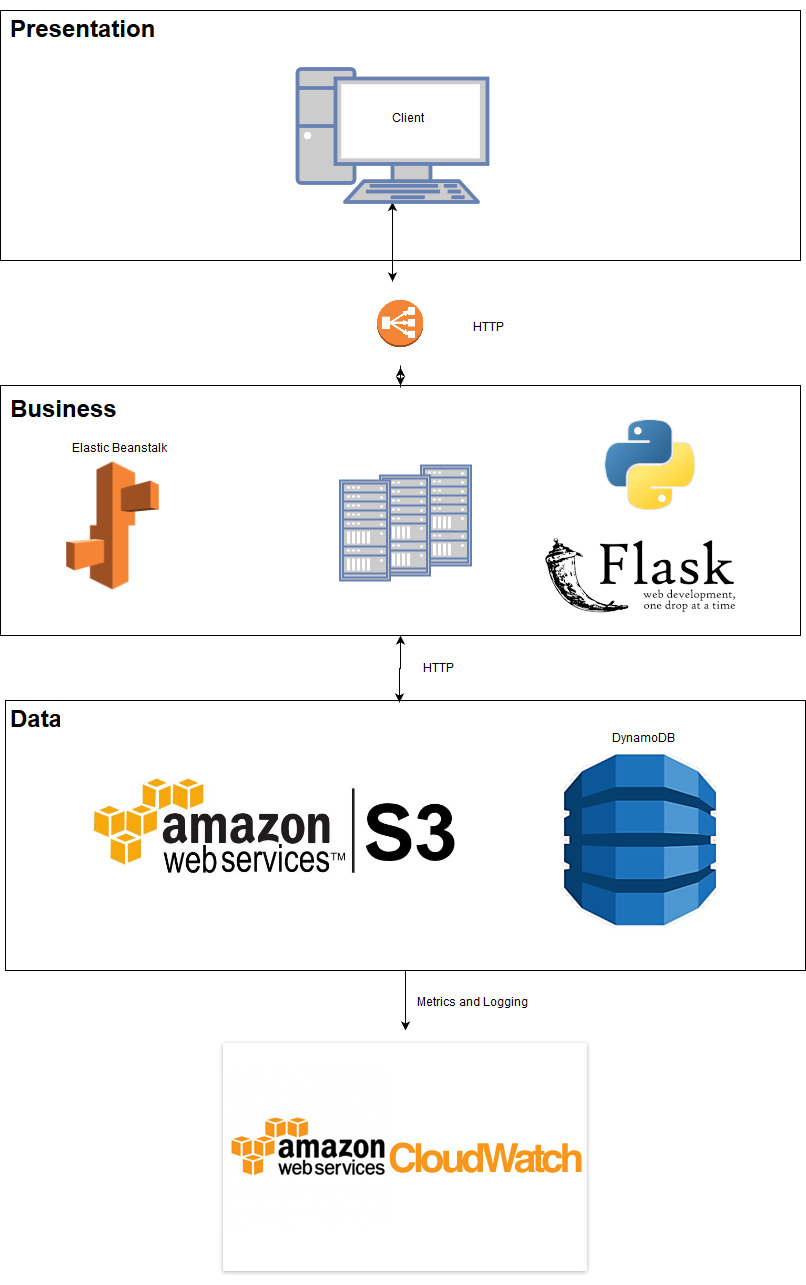
## index.html

The basic bones for the landing page. This includes the buttons, text fields, and test prompts.

## index.css

Styling sheet for the index.html page.

# Design



# How will it scale with load?

When the site is underload, AWS’ elastic load balancer will allow the traffic to be distributed across the different clients using them. In addition, this allows many instances to serve multiple people on different ports.

However, since there is only one DynamoDB and S3 storage for data storage, this will severely limit how many people can hit this at once. If one person is consistently adding or removing files, others will be impacted on how fast they can add, query, and remove items from the database.

# How do you monitor?

All of the services used benefit from another service called CloudWatch. This service allows you to create alarms, log events, and record metrics about your services. I use CloudWatch to keep track of errors and metrics. See images below.

A screenshot of a social media post

Description automatically generated

Figure 1: S3 usage and metrics

A screenshot of a computer

Description automatically generated

Figure 2: Elastic beanstalk usage and metrics

A screenshot of a computer

Description automatically generated

Figure 3: DyanmoDB usage and metrics

# Estimate of your SLA

DynamoDB

* Durability: 100% assumed by the data replication for fault tolerance
* Availability: 99.99%

S3

* Durability: 99.999999999%
* Availability: 99.99%

EC2

* Durability: 99.999999999
* Availability: 99.95%

Overall

* Durability: 100 \*99.999999999 \* 99.999999999 = 99.9999998
* Availability: 99.99\*99.99\*99.95 = 99.93