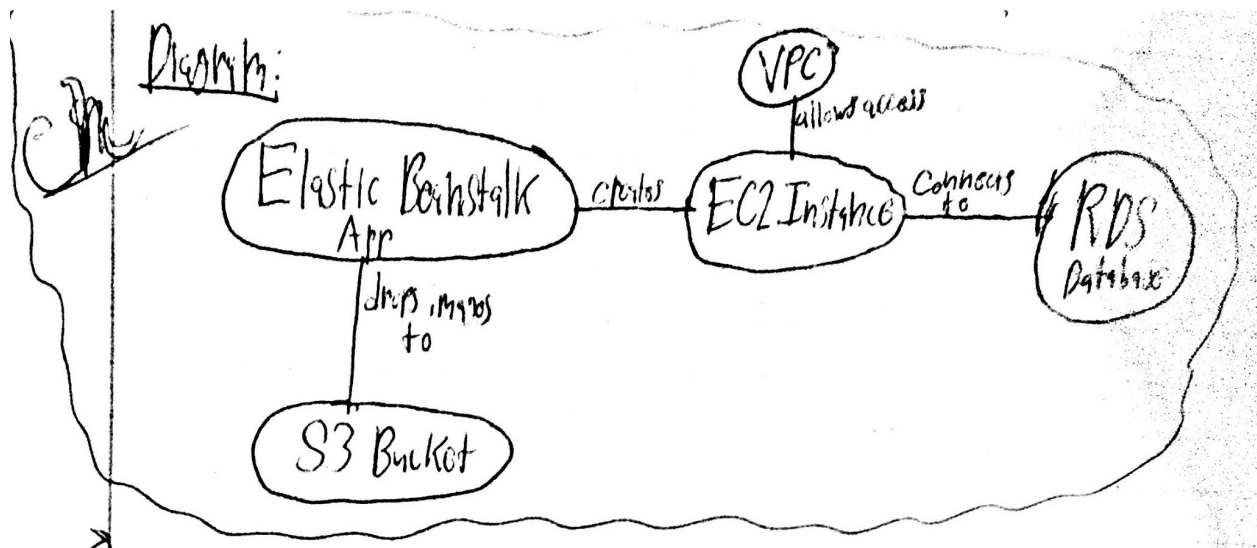


Carlos Mendoza  
CSCI 4452  
12/06/22  
Prof. Phani Vadrevu

### Assignment 3 Report/Documentation

For my AWS end-to-end web application, I decided to work as a group of one and aimed to build a straightforward weather app that would be uploaded to Elastic Beanstalk and return the weather in fahrenheit and a weather icon based on what the user searched for in a search bar. The search results, excluding the image, would be placed into a database made with RDS. Meanwhile, the associated image would be placed in a S3 bucket.

The web app would also display the previous search results and would act as a "search history." The items in this search history would be able to be deleted. Unfortunately, this previous search idea wasn't implemented into my final design due to time constraints and my overall minimal experience in both Python and AWS. Rest assured, however, the app still functions correctly based on my original idea in every other way using the following AWS services: Elastic Beanstalk, EC2, RDS, S3, and VPC. The following figure shows how the services connected with each other in order to deploy the app:

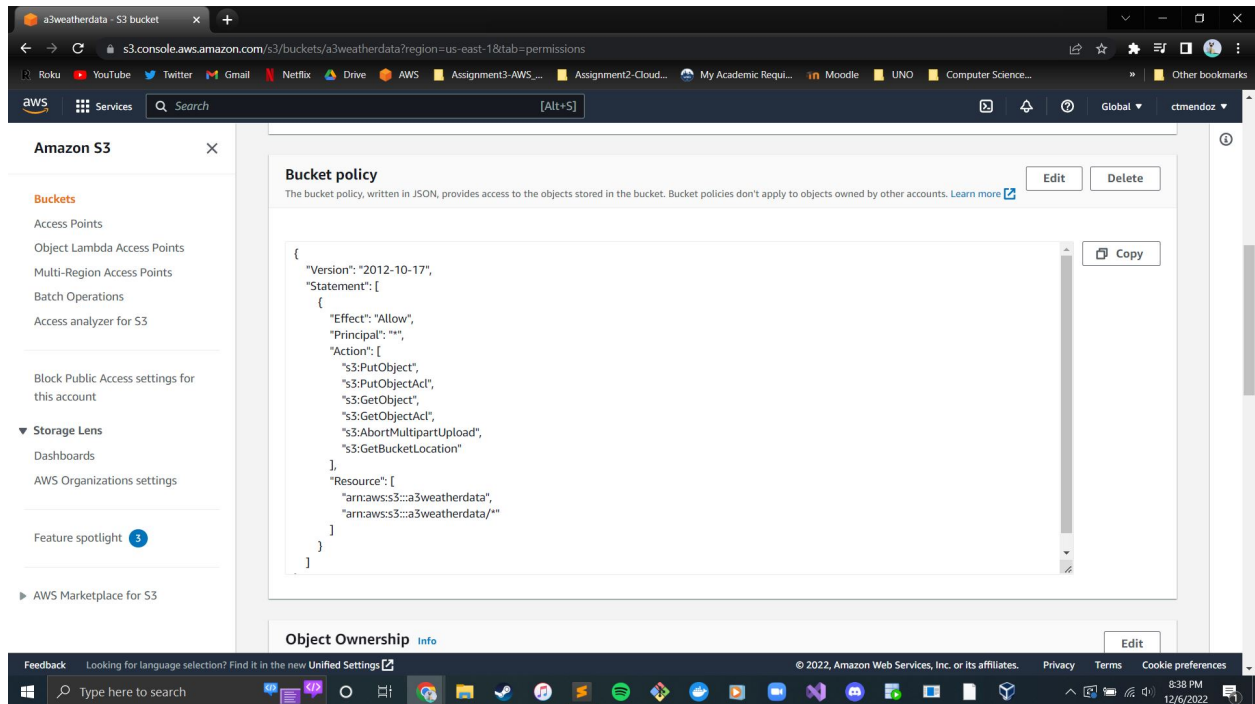


I roughly had 4 days to work on this assignment and as such wanted to create something simple yet unique and useful. Not only did this assignment help me obtain greater knowledge in the AWS services I used, but it also gave me a greater appreciation for cloud computing as a whole. For the remainder of this report, I plan to mention the general steps that I took in order to fully deploy my web app.

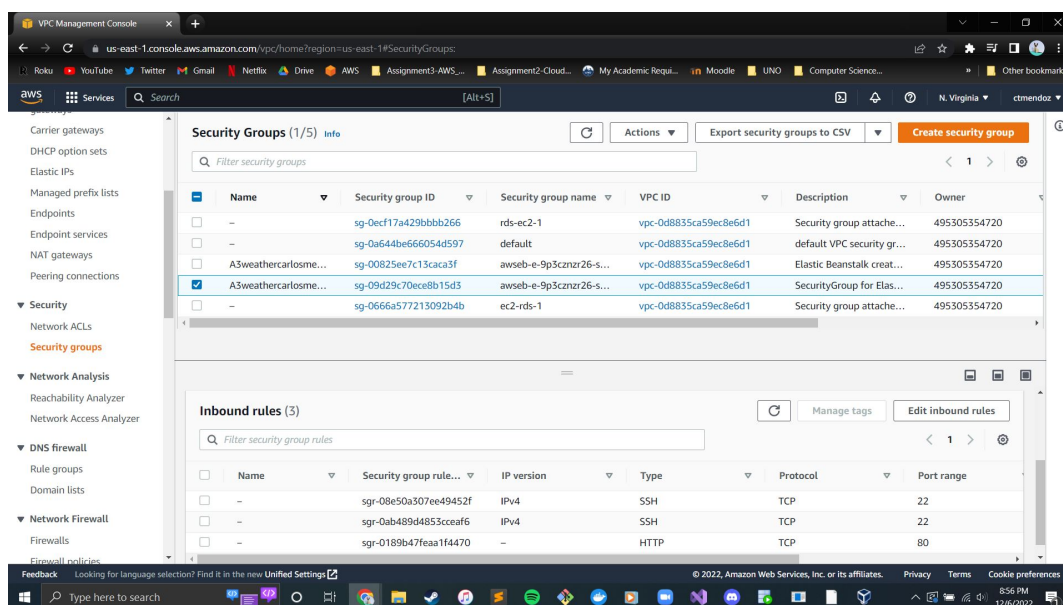
I began this assignment by creating the backbone of my app, the Python and HTML files. The implementation of my app-- and the idea of using OpenWeatherMap API-- was inspired by the following video: <https://youtu.be/WZNG8UomjSI>. Of course, the code in this video was in Javascript instead of Python, so I had to make several changes in order to get a working version of the app. Locally, the application.py and index.html files worked as expected. However, when I added any lines of code that were related to the database, my app wouldn't work. I understood why this wasn't the case as I decided to make sure that the app works locally before uploading it to Elastic Beanstalk and connecting it to my RDS database which I didn't set up at this point in time.

In order to have Elastic Beanstalk give me the OK health condition for my environment, I learned how to create a virtual environment with the following video: <https://youtu.be/dhHOzye-Rms>. I had to make a virtual environment in order to create a requirements.txt file which would tell Elastic Beanstalk what to install before running the app. The video also allowed me to create a .ebextensions folder with a python.config file that would essentially tell Elastic Beanstalk which file has to be executed in order to start the entire app.

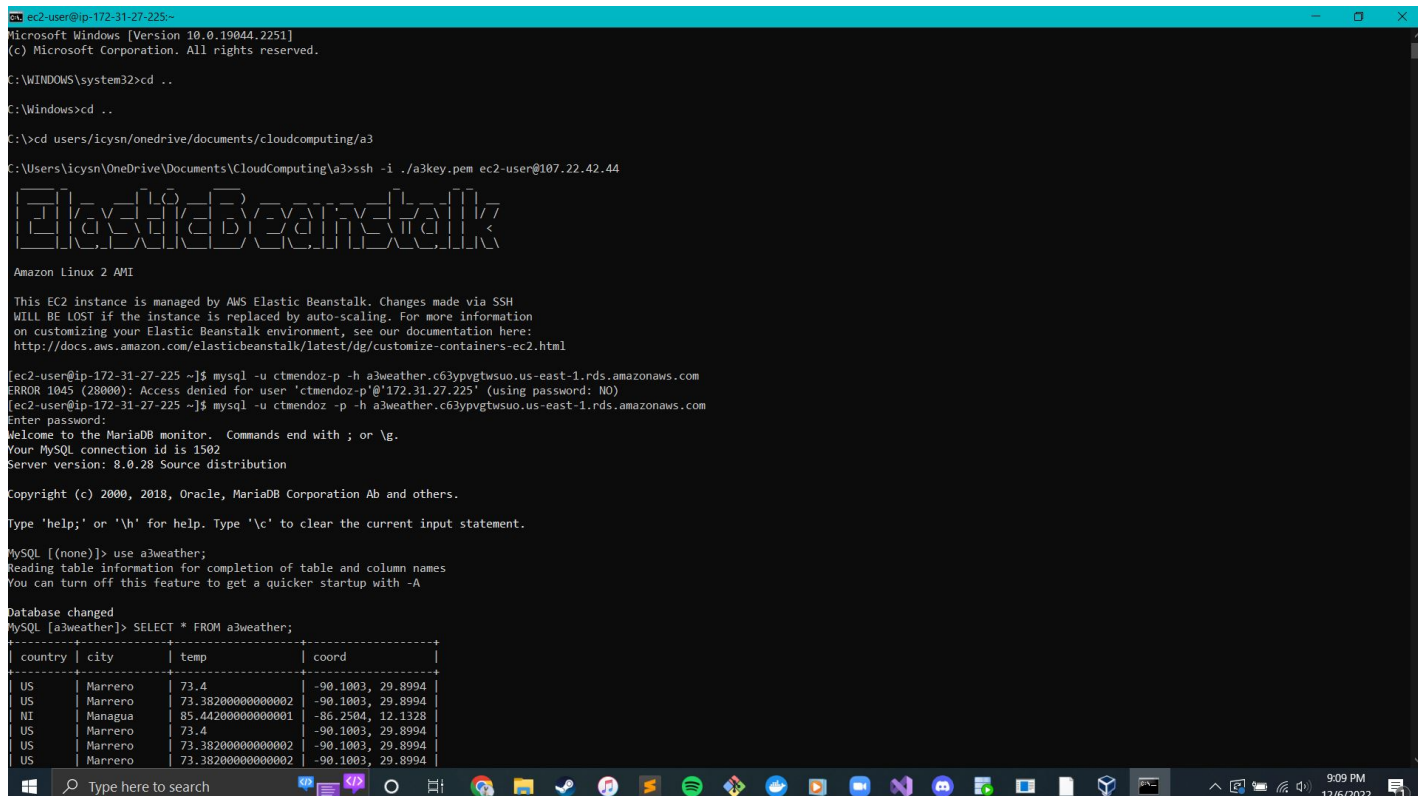
With my Elastic Beanstalk app up and running-- for the most part, that is-- I was then given an EC2 instance based on my app's environment. This would allow for me to connect my app with the other AWS services I planned on using for this project. I started with the S3 bucket which didn't involve the EC2 instance and was therefore relatively easy to set up. I already had the code set up for putting the image in the S3 bucket in my application.py file thanks to the boto3 module (see lines 104-115 in application.py). The big challenge with S3 was having to give my bucket public access. In order to ensure that my bucket remained safe, I only had public access unblocked when I was testing the app and I also plan to do it for when I demo my app in class. The following screenshot includes additional information in the form of the JSON bucket policy that I had to write in order to get my bucket to accept submissions:



Next, I decided to set up my RDS database which uses the MySQL engine. I then attempted to connect my Elastic Beanstalk app (via the EC2 server) with my RDS database by signing into the EC2 server. As I was getting ready to do this, though, I realized that I'm only used to signing into EC2 servers with a key pair. Because of this, I followed the video <https://youtu.be/7wvuuQGz728> in order to create a key-- a3key.pem-- in Elastic Beanstalk which, in turn, created the key in the EC2 instance associated with the app. I also used VPC in order to add an SSH inbound rule to the EC2 server's security group which you can see below:



I was now finally able to connect my EC2 instance with my RDS database. I did this by signing into the EC2 server and, while still in the server, accessed the MySQL database. I was also able to connect the two by placing all of the database's vital information (i.e. name, username, password, endpoint, etc.) in a file called config.py which application.py used (see lines 32-39). Furthermore, the pymysql module was used in order to execute and commit commands to my database through the application file. The act of accessing the database in the server can be seen in the following screenshot.



```
ec2-user@ip-172-31-27-225:~
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd ..
C:\Windows>cd ..
C:\>cd users\icysn\onedrive\documents\cloudcomputing/a3
C:\Users\icysn\OneDrive\Documents\CloudComputing\A3>ssh -i ./a3key.pem ec2-user@107.22.42.44

Elastic Beanstalk

Amazon Linux 2 AMI

This EC2 instance is managed by AWS Elastic Beanstalk. Changes made via SSH
WILL BE LOST if the instance is replaced by auto-scaling. For more information
on customizing your Elastic Beanstalk environment, see our documentation here:
http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/customize-containers-ec2.html

[ec2-user@ip-172-31-27-225 ~]$ mysql -u ctmdoz -p -h a3weather.c63ypvgtwsuo.us-east-1.rds.amazonaws.com
ERROR 1045 (28000): Access denied for user 'ctmdoz-p@172.31.27.225' (using password: NO)
[ec2-user@ip-172-31-27-225 ~]$ mysql -u ctmdoz -p -h a3weather.c63ypvgtwsuo.us-east-1.rds.amazonaws.com
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 1502
Server version: 8.0.28 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> use a3weather;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [a3weather]> SELECT * FROM a3weather;
+-----+-----+-----+-----+
| country | city      | temp      | coord      |
+-----+-----+-----+-----+
| US      | Marrero   | 73.4       | -90.1003, 29.8994 |
| US      | Marrero   | 73.38200000000002 | -90.1003, 29.8994 |
| NI      | Managua   | 85.44200000000001 | -86.2504, 12.1328 |
| US      | Marrero   | 73.4       | -90.1003, 29.8994 |
| US      | Marrero   | 73.38200000000002 | -90.1003, 29.8994 |
| US      | Marrero   | 73.38200000000002 | -90.1003, 29.8994 |
+-----+-----+-----+-----+
```

While all of these services correctly work in tandem with each other as planned, there were still a few problems and, ultimately, my original pitch didn't reflect my final product. To start, I should mention that the implementation of the database with my app did not go as planned since printing the database's contents proved to have no results; the app would work perfectly fine with the previous search code intact (see lines 74-75 in application.py and lines 52-58 in index.html), but the database info wouldn't show up on the web page for some reason--locally and through Elastic Beanstalk. Another significant issue with the deployment of my app was that the default city searched for when starting up the app, Marrero, would be implemented many times in the database throughout the first minute of deployment. This could easily be fixed by doing the "DELETE FROM a3weather WHERE city = 'Marrero'" command in MySQL, but I felt as though it still had to be addressed in this report. The final notable issue I encountered

was the fact that my app would become inaccessible through Elastic Beanstalk as the health condition would go from OK to Severe after about 5 minutes following the app's deployment. Thankfully, this issue can also be given a temporary fix by restarting the app server under "Actions."

Overall, even though I wasn't able to create my app the way I originally planned to, I was still successful in deploying my app in a way where all of the AWS services I planned on using were fully utilized and connected with each other in some way. Screenshots of the app working through the link provided by Elastic Beanstalk as well as the input being received by the S3 bucket and the RDS database can be seen below:

The screenshot displays a web browser window on the left and a terminal window on the right. The browser shows the 'AWS End-to-End Weather App - Assignment 3 CSCI 4452' created by Carlos Mendoza. The app's search results for 'NI' (Managua) show coordinates and temperature. The terminal window shows a series of MySQL commands and their outputs, including deleting records and selecting data from the 'a3weather' database.

**Browser Window:**

### AWS End-to-End Weather App - Assignment 3 CSCI 4452

created by Carlos Mendoza

Search

Country : NI  
City : Managua  
Coordinates : -86.2504, 12.1328  
Temperature (in Farenheit) : 72.842

Previous Searches:

**Terminal Window:**

```
MySQL [a3weather]> DELETE FROM a3weather WHERE country = 'NI';
Query OK, 2 rows affected (0.01 sec)

MySQL [a3weather]> DELETE FROM a3weather WHERE country = 'US';
Query OK, 3 rows affected (0.00 sec)

MySQL [a3weather]> SELECT * FROM a3weather;
+-----+-----+-----+-----+
| country | city | temp | coord |
+-----+-----+-----+-----+
| NI | Managua | 72.842 | -86.2504, 12.1328 |
| US | Marrero | 70.736000000000008 | -90.1003, 29.8994 |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

MySQL [a3weather]> SELECT * FROM a3weather;
+-----+-----+-----+-----+
| country | city | temp | coord |
+-----+-----+-----+-----+
| NI | Managua | 72.842 | -86.2504, 12.1328 |
| US | Marrero | 70.736000000000008 | -90.1003, 29.8994 |
| US | Marrero | 70.790000000000002 | -90.1003, 29.8994 |
+-----+-----+-----+-----+
3 rows in set (0.01 sec)

MySQL [a3weather]>
```

Amazon S3 console interface showing a list of objects in a bucket named 'a3weatherdata'.

**Objects (13)**

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Actions: Copy S3 URI, Copy URL, Download, Open, Delete, Actions, Create folder, Upload

Find objects by prefix

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	<a href="#">weathermarrero_image_file.png</a>	png	December 6, 2022, 21:37:19 (UTC-06:00)	1.0 KB	Standard
<input type="checkbox"/>	<a href="#">weathermanagua_image_file.png</a>	png	December 6, 2022, 21:35:49 (UTC-06:00)	867.0 B	Standard
<input type="checkbox"/>	<a href="#">weatherManagua_image_file.png</a>	png	December 6, 2022, 21:35:22 (UTC-06:00)	867.0 B	Standard
<input type="checkbox"/>	<a href="#">weatherBoring_image_file.png</a>	png	December 6, 2022, 21:32:14 (UTC-06:00)	401.0 B	Standard
<input type="checkbox"/>	<a href="#">weatheroakland_image_file.png</a>	png	December 6, 2022, 16:46:02 (UTC-06:00)	1.0 KB	Standard
<input type="checkbox"/>	<a href="#">weathersanta+claus_image_file.png</a>	png	December 6, 2022, 16:45:35 (UTC-06:00)	401.0 B	Standard
<input type="checkbox"/>	<a href="#">weatherboring_image_file.png</a>	png	December 6, 2022, 16:45:12 (UTC-06:00)	1.0 KB	Standard
<input type="checkbox"/>	<a href="#">weatherkyoto_image_file.png</a>	png	December 3, 2022, 16:39:43 (UTC-06:00)	1.0 KB	Standard
<input type="checkbox"/>	<a href="#">weatherlafayette_image_file.png</a>	png	December 3, 2022, 16:34:37 (UTC-06:00)	432.0 B	Standard
<input type="checkbox"/>	<a href="#">weathermiami_image_file.png</a>	png	December 3, 2022, 16:26:17 (UTC-06:00)	432.0 B	Standard
<input type="checkbox"/>	<a href="#">weatherdestin_image_file.png</a>	png	December 3, 2022, 16:23:00 (UTC-06:00)	1.0 KB	Standard
<input type="checkbox"/>	<a href="#">weatherhouma_image_file.png</a>	png	December 3, 2022, 16:09:27 (UTC-06:00)	1.3 KB	Standard
<input type="checkbox"/>	<a href="#">weathernew+york_image_file.png</a>	png	December 3, 2022, 15:43:37 (UTC-06:00)	1.0 KB	Standard

Feedback Looking for language selection? Find it in the new Unified Settings

© 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

9:37 PM 12/6/2022