CS – 499 Capstone: Milestone Two

Enhancement One: Software Design/Engineering

Milton Francisco

Southern New Hampshire University

March 17, 2025

Briefly describe the artifact. What is it? When was it created?

This artifact is a data dashboard that was originally created in an Apporto virtual environment. It was created last year as a project for CS-340, which is a Client/Server Development class. The dashboard used a provided instance of MongoDB, Python middleware to interact with the database, and Jupyter Notebook with Plotly Dash to render the user interface. The dashboard aimed to present the data in a user-friendly way and provided additional information such as graphs and maps to represent selected data.

Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in software development? How was the artifact improved?

I selected this item because it is currently unable to achieve its purpose and has a clear path for improvement. Additionally, the enhancements will highlight my ability to work with cloud computing, enhance application accessibility and implement best security practices.

The design of this artifact is improved by running MongoDB in the cloud, refactoring the Python middleware for the new connection, and modifying the Jupyter Notebook to work in a different local environment. Originally, the database was only able to be accessed from a specific Apporto environment. Now, the database is now able to be remotely accessed, if the correct credentials are presented. Additionally, the application can now run within a local Python environment, showcasing my ability to enhance an application's accessibility. Finally, enforcing strict EC2 security group rules and implementing role-based access control within MongoDB helps ensure only authorized users can access data, significantly reducing potential vulnerabilities.

Additionally, my code review highlighted multiple areas for improvement within this artifact and the first iteration of changes have been implemented. This includes cleaning the script to remove unnecessary comments, stubs that are no longer needed and updating logic errors. During enhancement three, this artifact will be further improved when the user interface is refactored to use the newer Dash library in Python.

Did you meet the course outcomes you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

This enhancement exceeds the outcomes I planned in my proposal. The 1st outcome, *(Employ strategies for building collaborative environments that enable diverse audiences to support organizational decision making in the field of computer science)*, is achieved by removing the dependency on an Apporto local environment and making the application more accessible to a wider audience.

The 4th outcome, *(Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals),* is demonstrated through using the design pattern of a cloud-based database, middleware to access it and a dashboard for the end user.

Behind the scenes, the 5th outcome, *(Develop a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources),* is demonstrated through the EC2 instance security and MongoDB configuration. By enforcing strict EC2 groups

One potential update I have for my plan is to simplify the entire setup process by using Docker. This could allow the application to be run without worrying about versions or setting up a virtual environment manually. Although I did not have enough time to implement it this week, it is on my radar for the third enhancement. Additionally, it will be more efficient to implement Docker after the dashboard has been refactored to use the newer Dash library in a Python file.

Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?

I learned a significant amount about AWS and expanded my current knowledge on the other parts of the application I worked on. Before this, I had not worked with AWS, but there is extensive documentation available. When I used Apporto in my previous classes, it did not feel right because some parts were already set up for us. I understand that this is more than likely to help students focus on the specific objectives for a class, but I like knowing how the entire application works. In CS-340 we were given the correct credentials to use the database and some of the important code was provided. Despite venturing into new territory, I enjoyed researching the correct methods and implementations and extensively researched multiple official documentation sites.

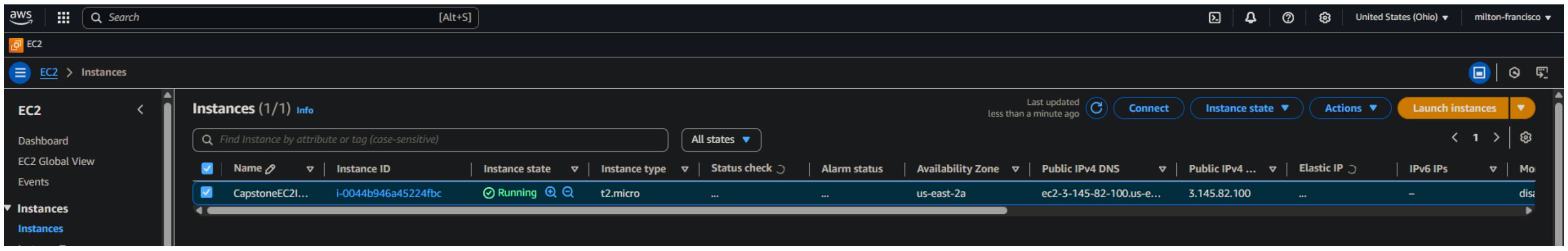
I faced a significant challenge in this enhancement, as the Solid-State Drive I used for my Operating System failed. This caused me to lose time and quite a few important files. Luckily, I was able to recover some things, but the most important recovery was my RSA private key I use to access the created EC2 instance. Luckily, I am familiar with the process now, so if I had lost my private key, it would not have been too detrimental. I was not expecting to learn about data recovery in this course, but I now know more!

Configuring the EC2 security was also insightful as it required thought into how it will be used for this artifact. After contemplation, I realized that I am the sole person who should be able to access the EC2 instance, via PuTTY SSH connection, so I limited the inbound traffic to SSH for myself and allowed a port for the MongoDB connection. I also configured MongoDB with separate users, one for the admin with root access (me), and one for the application user with ‘read/write’ access to interact with the applicable database.

Additionally, refactoring the Jupyter Notebook to work in my local environment proved difficult and required quite a bit of troubleshooting to make it work as it did in Apporto. Ultimately, I had to modify database indices because the file imported differently than it did in the original project. I also updated the map view to account for this difference, along with improvements by removing hard-coded column numbers.

Enhancement Pictures

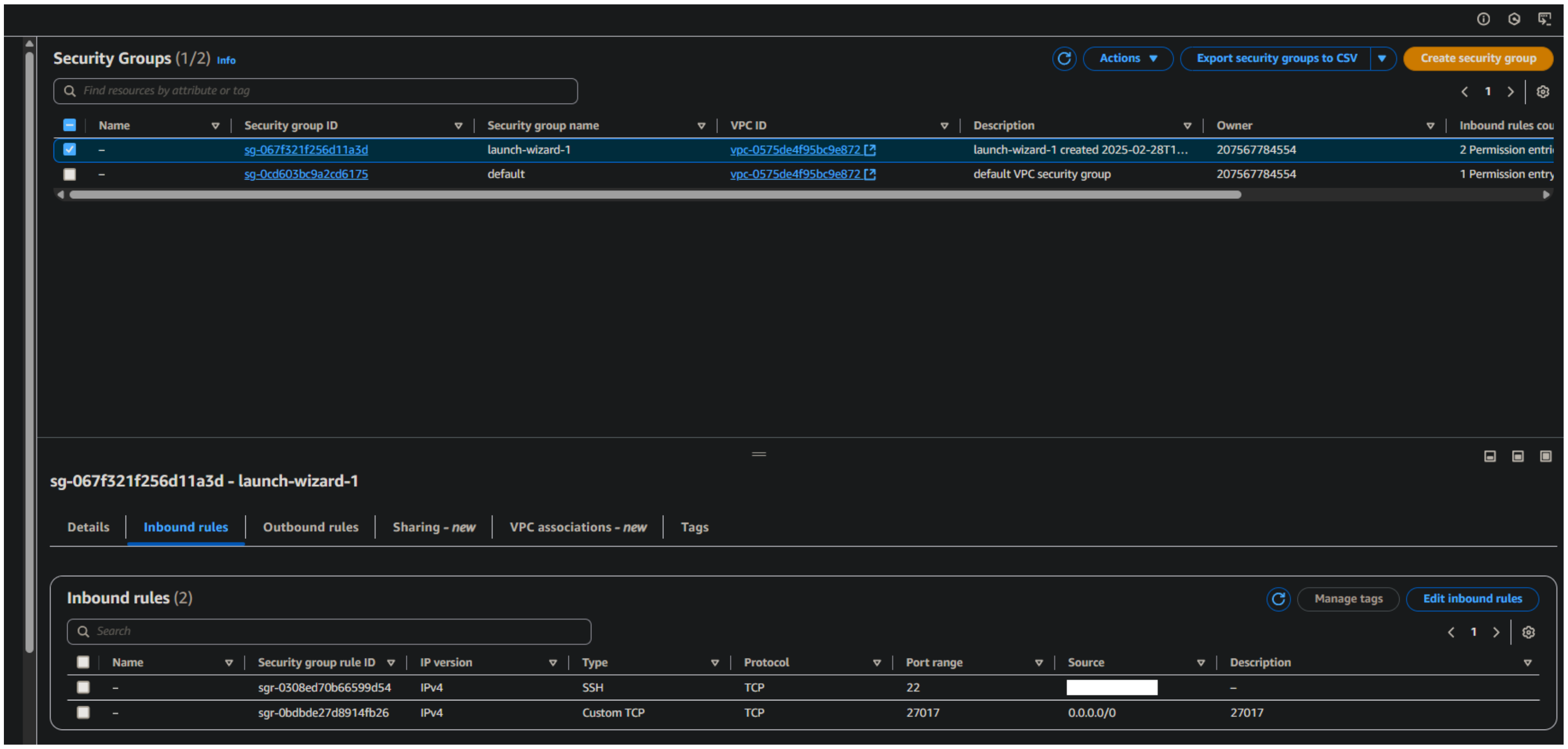
EC2 Instance Running on AWS:



Instances (1/1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Mo
CapstoneEC2I...	i-0044b946a45224fbc	Running	t2.micro	...	...	us-east-2a	ec2-3-145-82-100.us-e...	3.145.82.100	...	–	dis

EC2 Security Groups: (Sensitive info removed)



Security Groups (1/2) Info

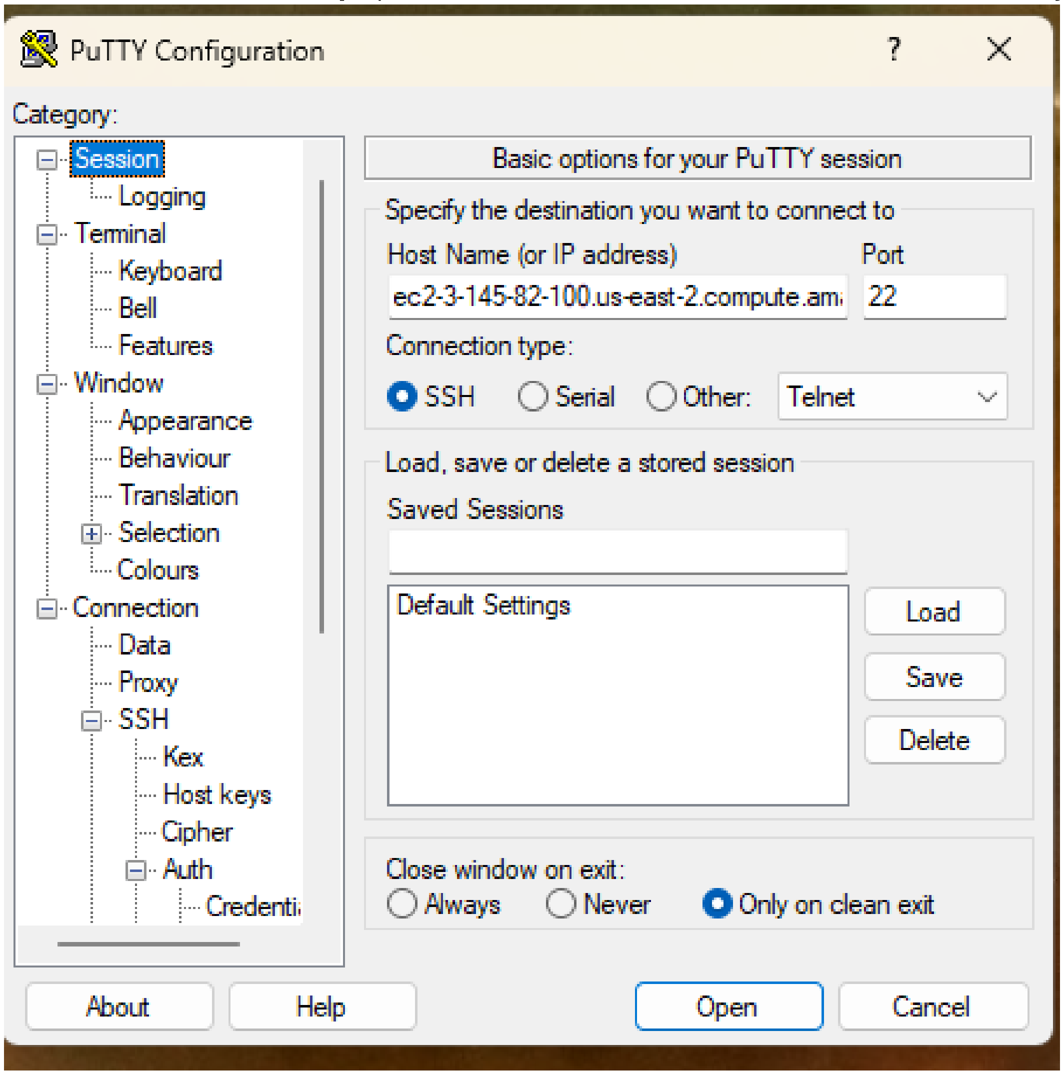
Name	Security group ID	Security group name	VPC ID	Description	Owner	Inbound rules cou
–	sg-067f321f256d11a3d	launch-wizard-1	vpc-0575de4f95bc9e872	launch-wizard-1 created 2025-02-28T1...	207567784554	2 Permission entri
–	sg-0cd603bc9a2cd6175	default	vpc-0575de4f95bc9e872	default VPC security group	207567784554	1 Permission entry

sg-067f321f256d11a3d - launch-wizard-1

Inbound rules (2)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
–	sgr-0308ed70b66599d54	IPv4	SSH	TCP	22	[redacted]	–
–	sgr-0bdbde27d8914fb26	IPv4	Custom TCP	TCP	27017	0.0.0.0/0	27017

PuTTY connection Setup: (Connection>SSH>Auth>Credentials: Contains my private key)



Specify the destination you want to connect to

Host Name (or IP address): ec2-3-145-82-100.us-east-2.compute.am

Port: 22

Connection type: [✓ SSH] Serial Other: Telnet

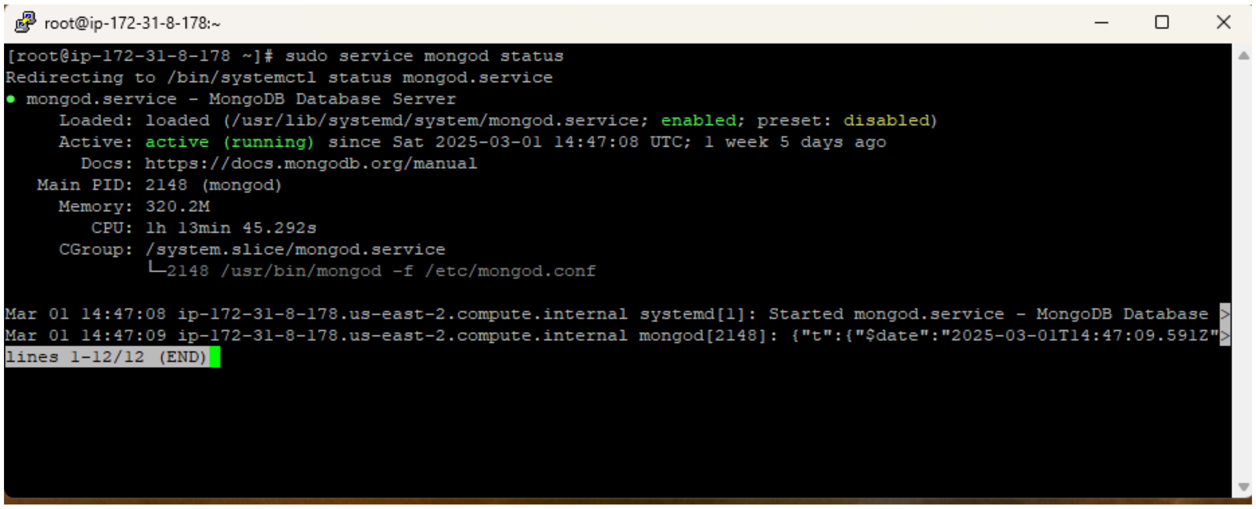
Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

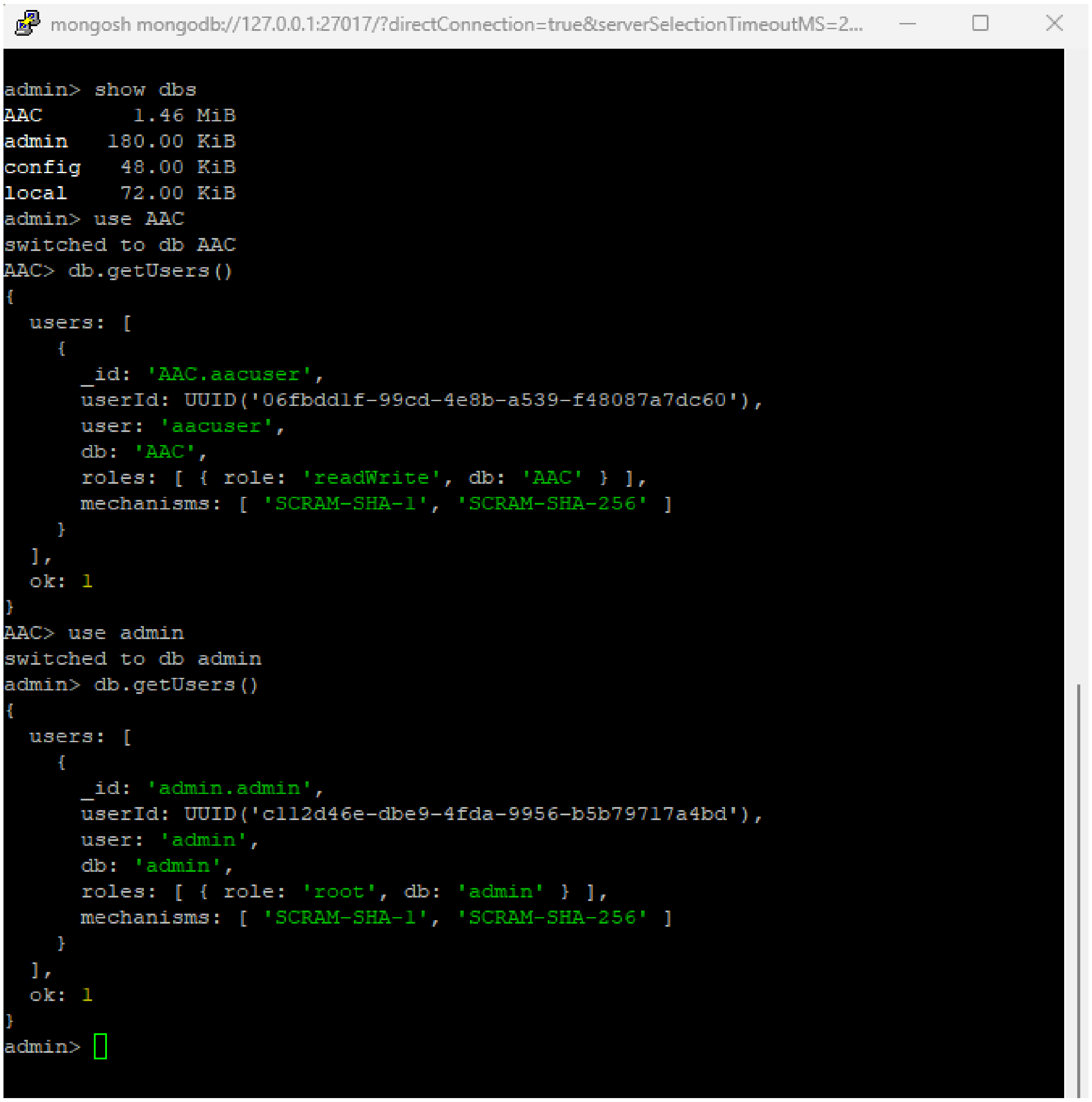
MongoDB Running in EC2 Instance (with authentication enabled):



```
[root@ip-172-31-8-178 ~]# sudo service mongod status
Redirecting to /bin/systemctl status mongod.service
● mongod.service - MongoDB Database Server
     Loaded: loaded (/usr/lib/systemd/system/mongod.service; enabled; preset: disabled)
     Active: active (running) since Sat 2025-03-01 14:47:08 UTC; 1 week 5 days ago
       Docs: https://docs.mongodb.org/manual
   Main PID: 2148 (mongod)
     Memory: 320.2M
        CPU: 1h 13min 45.292s
     CGroup: /system.slice/mongod.service
             └─2148 /usr/bin/mongod -f /etc/mongod.conf

Mar 01 14:47:08 ip-172-31-8-178.us-east-2.compute.internal systemd[1]: Started mongod.service - MongoDB Database >
Mar 01 14:47:09 ip-172-31-8-178.us-east-2.compute.internal mongod[2148]: {"t":{"$date":"2025-03-01T14:47:09.591Z">
lines 1-12/12 (END)
```

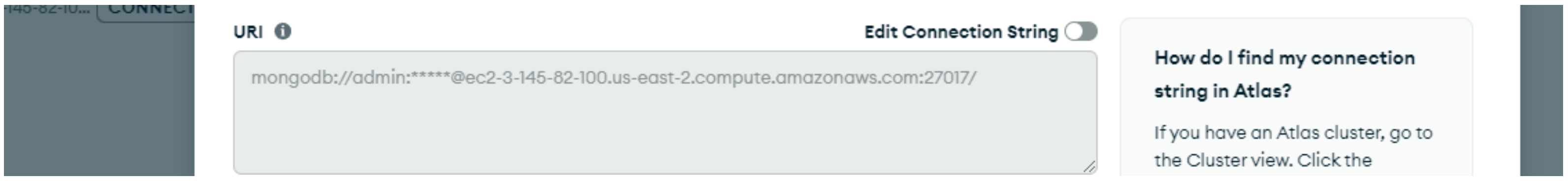
Role-based Access Control in MongoDB: (Logged in as admin)



mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2...

```
admin> show dbs
AAC        1.46 MiB
admin    180.00 KiB
config    48.00 KiB
local     72.00 KiB
admin> use AAC
switched to db AAC
AAC> db.getUsers()
{
  users: [
    {
      _id: 'AAC.aacuser',
      userId: UUID('06fbdd1f-99cd-4e8b-a539-f48087a7dc60'),
      user: 'aacuser',
      db: 'AAC',
      roles: [ { role: 'readWrite', db: 'AAC' } ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    }
  ],
  ok: 1
}
AAC> use admin
switched to db admin
admin> db.getUsers()
{
  users: [
    {
      _id: 'admin.admin',
      userId: UUID('c112d46e-dbe9-4fda-9956-b5b79717a4bd'),
      user: 'admin',
      db: 'admin',
      roles: [ { role: 'root', db: 'admin' } ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    }
  ],
  ok: 1
}
admin> 
```

MongoDB Compass Connection:

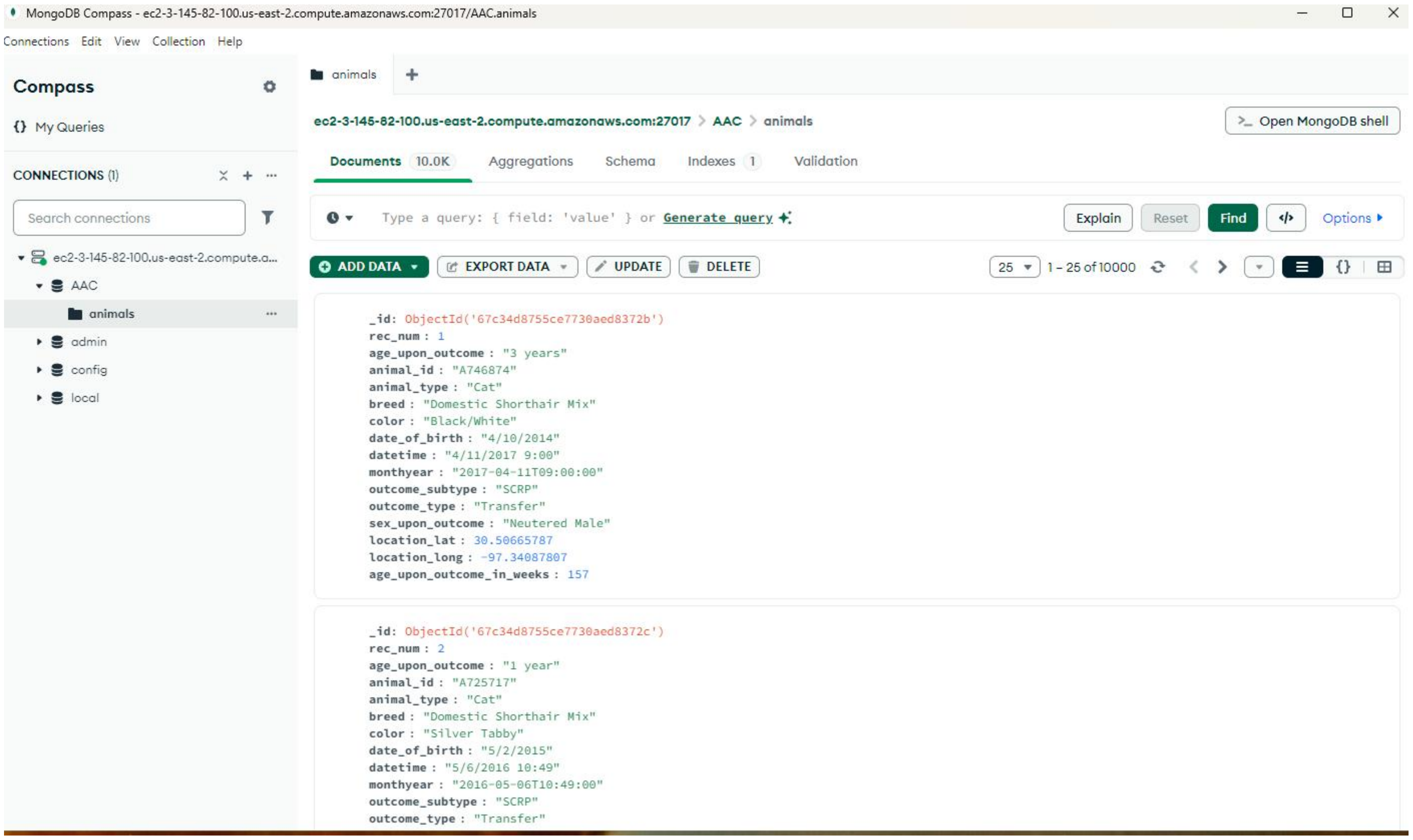


```
mongodb://admin:*****@ec2-3-145-82-100.us-east-2.compute.amazonaws.com:27017/
```

How do I find my connection string in Atlas?

If you have an Atlas cluster, go to the Cluster view. Click the

MongoDB Compass Connection & Data Import:

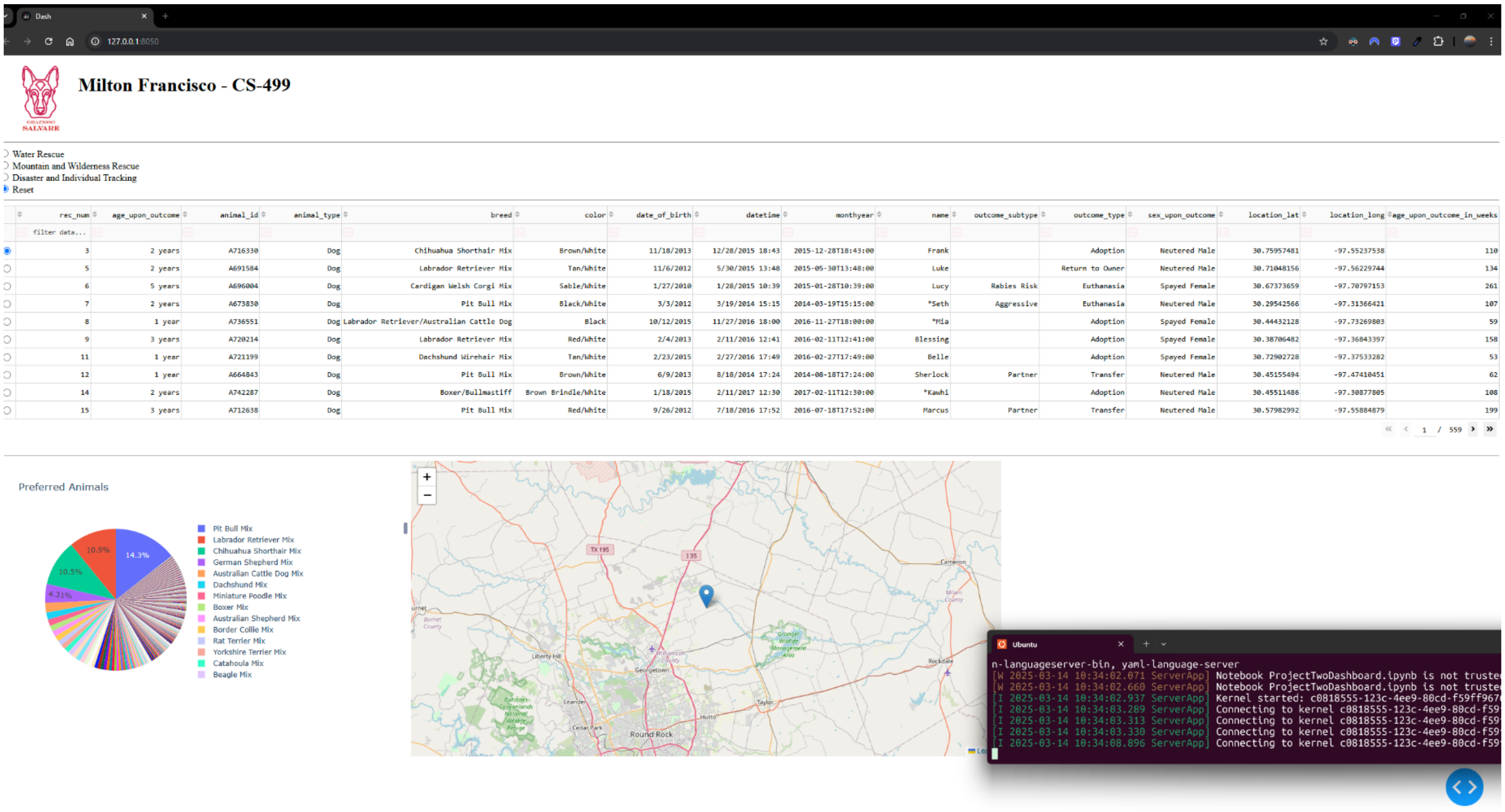


```
_id: ObjectId('67c34d8755ce7730aed8372b')
rec_num : 1
age_upon_outcome : "3 years"
animal_id : "A746874"
animal_type : "Cat"
breed : "Domestic Shorthair Mix"
color : "Black/White"
date_of_birth : "4/10/2014"
datetime : "4/11/2017 9:00"
monthyear : "2017-04-11T09:00:00"
outcome_subtype : "SCRP"
outcome_type : "Transfer"
sex_upon_outcome : "Neutered Male"
location_lat : 30.50665787
location_long : -97.34087807
age_upon_outcome_in_weeks : 157
```

```
_id: ObjectId('67c34d8755ce7730aed8372c')
rec_num : 2
age_upon_outcome : "1 year"
animal_id : "A725717"
animal_type : "Cat"
breed : "Domestic Shorthair Mix"
color : "Silver Tabby"
date_of_birth : "5/2/2015"
datetime : "5/6/2016 10:49"
monthyear : "2016-05-06T10:49:00"
outcome_subtype : "SCRP"
outcome_type : "Transfer"
```

App Running on local Jupyter Notebook:

(Instructions to run are included in the enhancement folder. (‘HowToRunDashboard.docx’) After creating the documentation to run this, I realized Docker could make this much easier, so I am looking forward to implementing that for enhancement 3)



Milton Francisco - CS-499

Water Rescue
Mountain and Wilderness Rescue
Disaster and Individual Tracking
[✓ Reset]

	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype	outcome_type	sex_upon_outcome	location_lat	location_long	age_upon_outcome_in_weeks
	filter data...															
●	3	2 years	A716330	Dog	Chihuahua Shorthair Mix	Brown/White	11/18/2013	12/28/2015 18:43	2015-12-28T18:43:00	Frank		Adoption	Neutered Male	30.75957481	-97.55237538	110
○	5	2 years	A691584	Dog	Labrador Retriever Mix	Tan/White	11/6/2012	5/30/2015 13:48	2015-05-30T13:48:00	Luke		Return to Owner	Neutered Male	30.71048156	-97.56229744	134
○	6	5 years	A696004	Dog	Cardigan Welsh Corgi Mix	Sable/White	1/27/2010	1/28/2015 10:39	2015-01-28T10:39:00	Lucy	Rabies Risk	Euthanasia	Spayed Female	30.67373659	-97.70797153	261
○	7	2 years	A673830	Dog	Pit Bull Mix	Black/White	3/3/2012	3/19/2014 15:15	2014-03-19T15:15:00	*Seth	Aggressive	Euthanasia	Neutered Male	30.29542566	-97.31366421	107
○	8	1 year	A736551	Dog	Labrador Retriever/Australian Cattle Dog	Black	10/12/2015	11/27/2016 18:00	2016-11-27T18:00:00	*Mia		Adoption	Spayed Female	30.44432128	-97.73269803	59
○	9	3 years	A720214	Dog	Labrador Retriever Mix	Red/White	2/4/2013	2/11/2016 12:41	2016-02-11T12:41:00	Blessing		Adoption	Spayed Female	30.38706482	-97.36843397	158
○	11	1 year	A721199	Dog	Dachshund Wirehair Mix	Tan/White	2/23/2015	2/27/2016 17:49	2016-02-27T17:49:00	Belle		Adoption	Spayed Female	30.72902728	-97.37533282	53
○	12	1 year	A664843	Dog	Pit Bull Mix	Brown/White	6/9/2013	8/18/2014 17:24	2014-08-18T17:24:00	Sherlock	Partner	Transfer	Neutered Male	30.45155494	-97.47410451	62
○	14	2 years	A742287	Dog	Boxer/Bullmastiff	Brown Brindle/White	1/18/2015	2/11/2017 12:30	2017-02-11T12:30:00	*Kawhi		Adoption	Neutered Male	30.45511486	-97.30877805	108
○	15	3 years	A712638	Dog	Pit Bull Mix	Red/White	9/26/2012	7/18/2016 17:52	2016-07-18T17:52:00	Marcus	Partner	Transfer	Neutered Male	30.57982992	-97.55884879	199

1 / 559

```
n-languageserver-bin, yaml-language-server
[W 2025-03-14 10:34:02.071 ServerApp] Notebook ProjectTwoDashboard.ipynb is not truste
[W 2025-03-14 10:34:02.660 ServerApp] Notebook ProjectTwoDashboard.ipynb is not truste
[I 2025-03-14 10:34:02.937 ServerApp] Kernel started: c0818555-123c-4ee9-80cd-f59ff967
[I 2025-03-14 10:34:03.289 ServerApp] Connecting to kernel c0818555-123c-4ee9-80cd-f59
[I 2025-03-14 10:34:03.313 ServerApp] Connecting to kernel c0818555-123c-4ee9-80cd-f59
[I 2025-03-14 10:34:03.330 ServerApp] Connecting to kernel c0818555-123c-4ee9-80cd-f59
[I 2025-03-14 10:34:08.896 ServerApp] Connecting to kernel c0818555-123c-4ee9-80cd-f59
```