

Big Data System – MongoDB Lab Sheet:4

MongoDB Data Manipulation with Python

1. Objective

Students should be able to

- A. Connect MongoDB Atlas to python
- B. Perform CRUD operations on MongoDB using python

2. Introduction

Like relational databases, NoSQL databases such as MongoDB can also be connected to a programming language and access and manipulate the data using programming language. In this lab demonstration, we will connect MongoDB Atlas with python and will try to access and manipulate the data stored in Mongo Atlas.

MongoDB Atlas

MongoDB Atlas is a fully managed cloud based database service by the same people that build MongoDB. It simplifies deploying and managing your databases while offering the versatility you need to build resilient and performant global applications on the cloud providers of your choice.

Python

Python, a dynamically typed language, has comprehensive support for common data manipulation and processing tasks. Python's native dictionary and list data types make it suitable for manipulating JSON documents. PyMongo, the standard MongoDB driver library for Python, is easy to use and offers an intuitive API for accessing databases, collections, and documents.

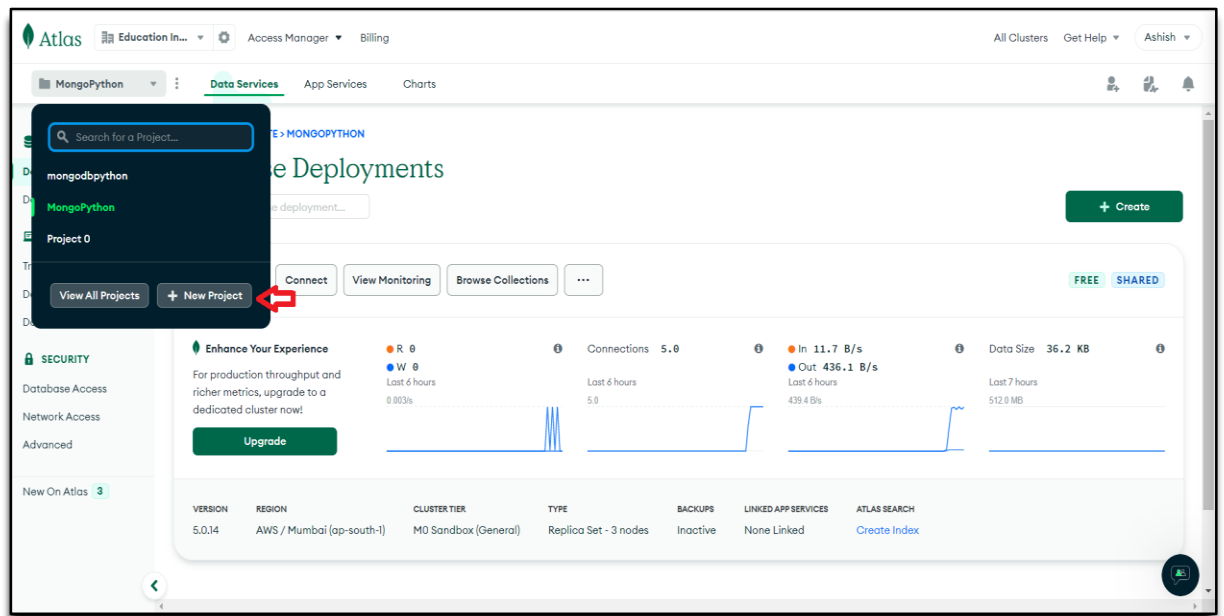
Objects retrieved from MongoDB through PyMongo are compatible with dictionaries and lists, so we can easily manipulate, iterate, and print them.

3. Steps to be performed

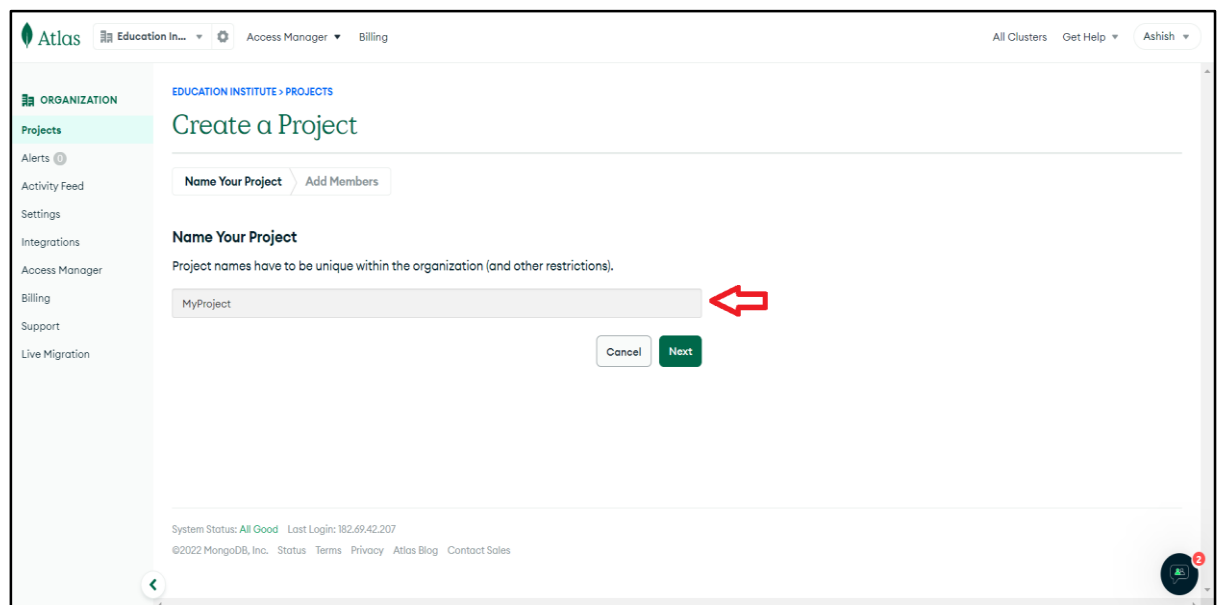
Create an account on cloud.mongodb.com

Setup a cluster by following the below steps

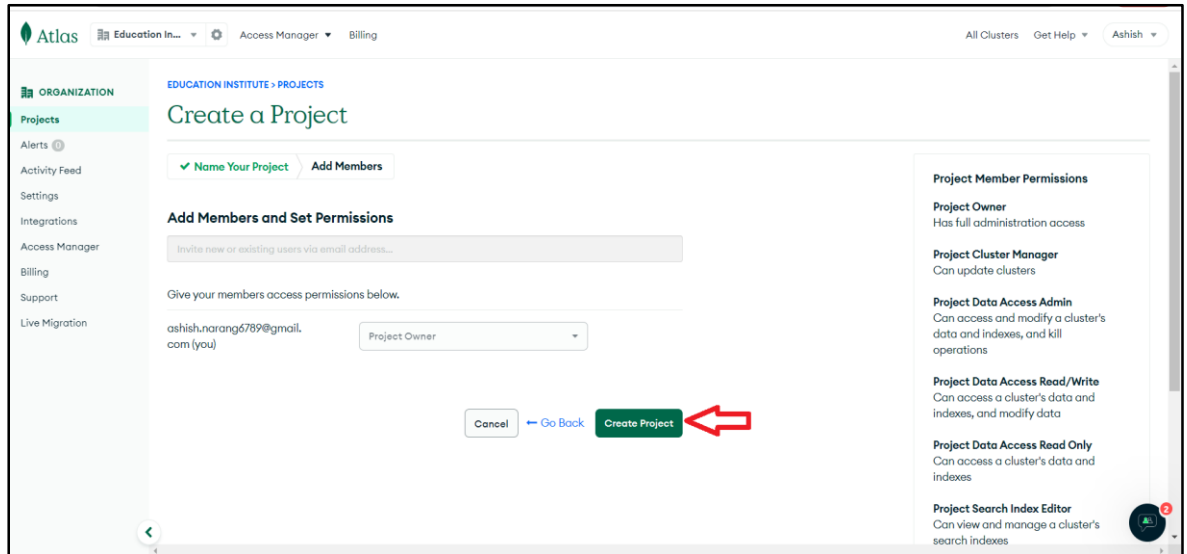
a) Create a new project by clicking on the drop-down on top left corner as follows



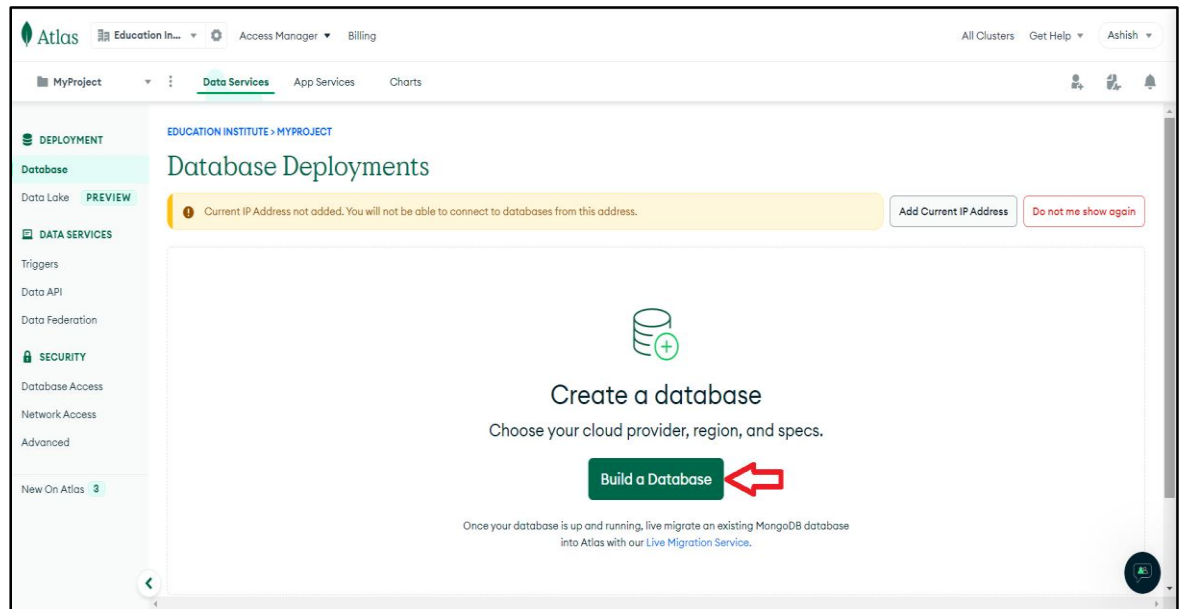
b) Give a name to your project and click next



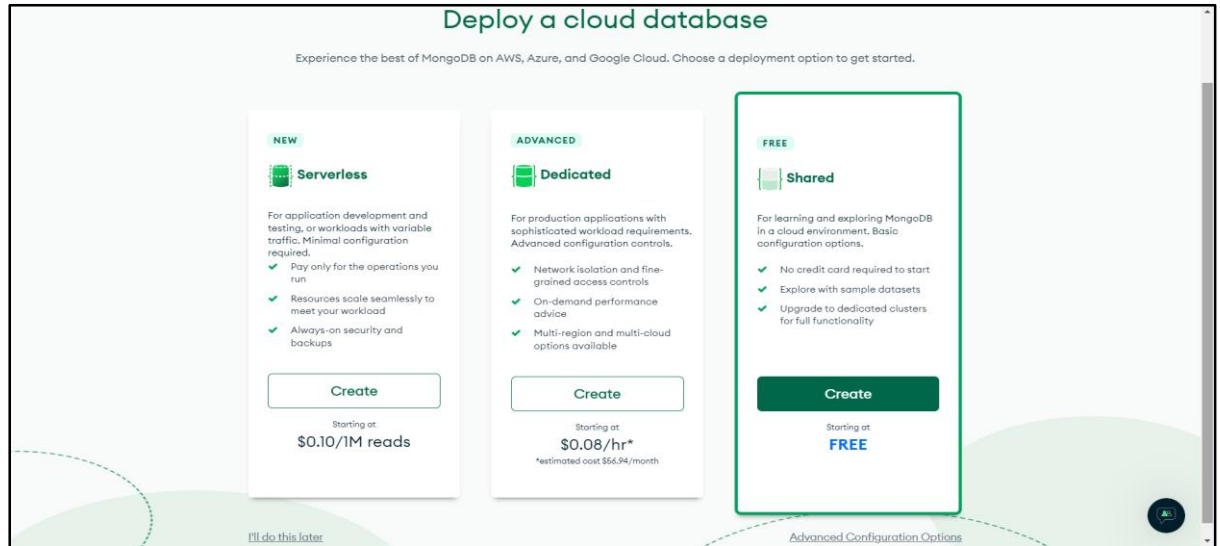
c) It displays owner's email id. Verify your email id and click on create project



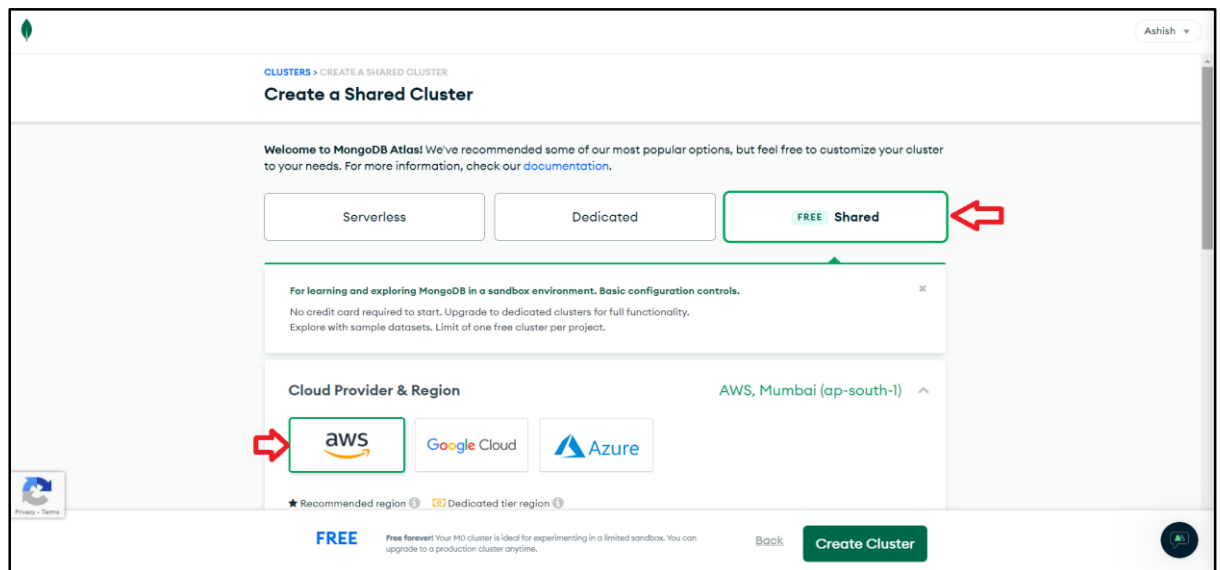
d) Build the database by clicking on build database button



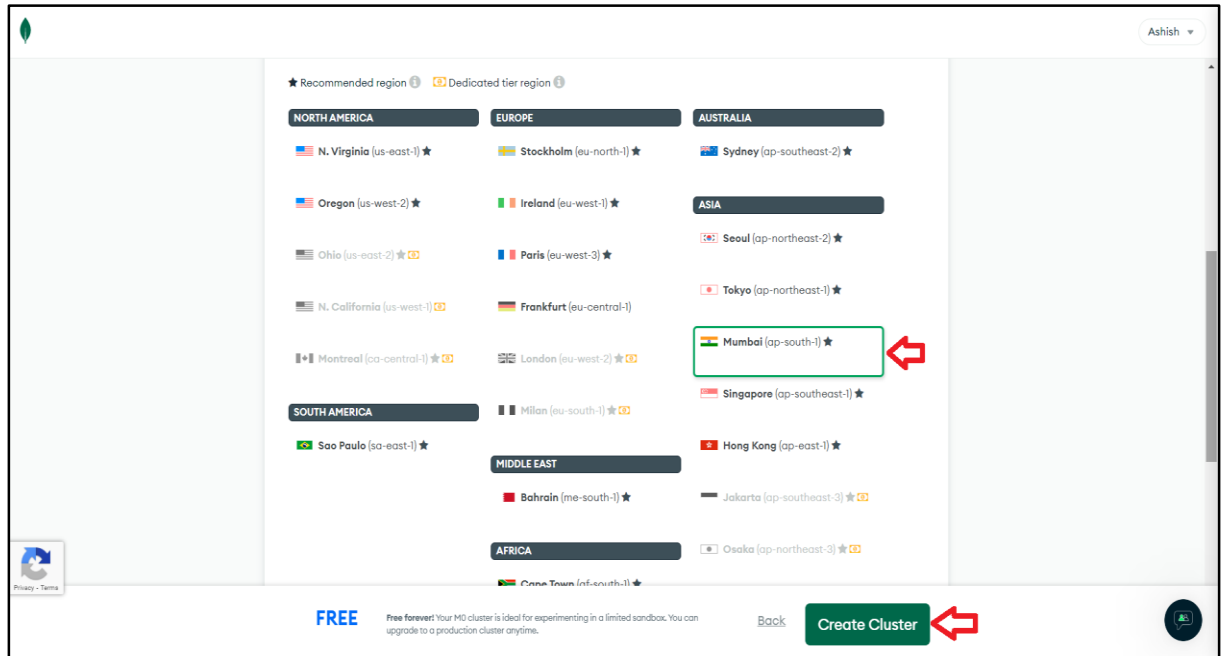
e) Choose the shared(free) option to deploy the database



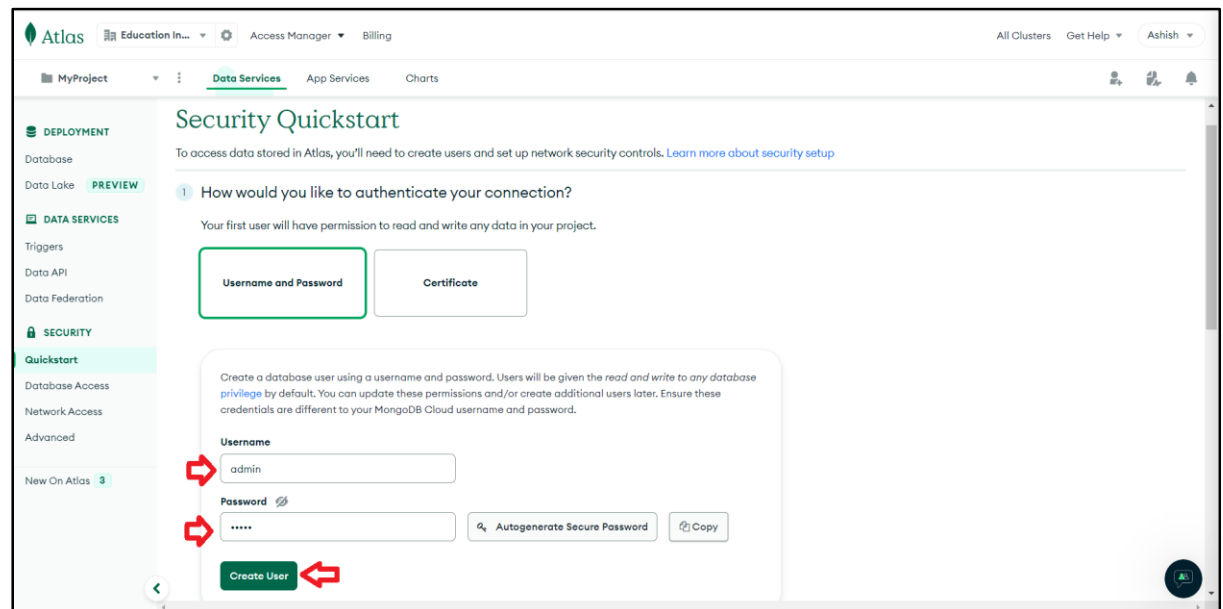
f) Choose AWS as cloud provider



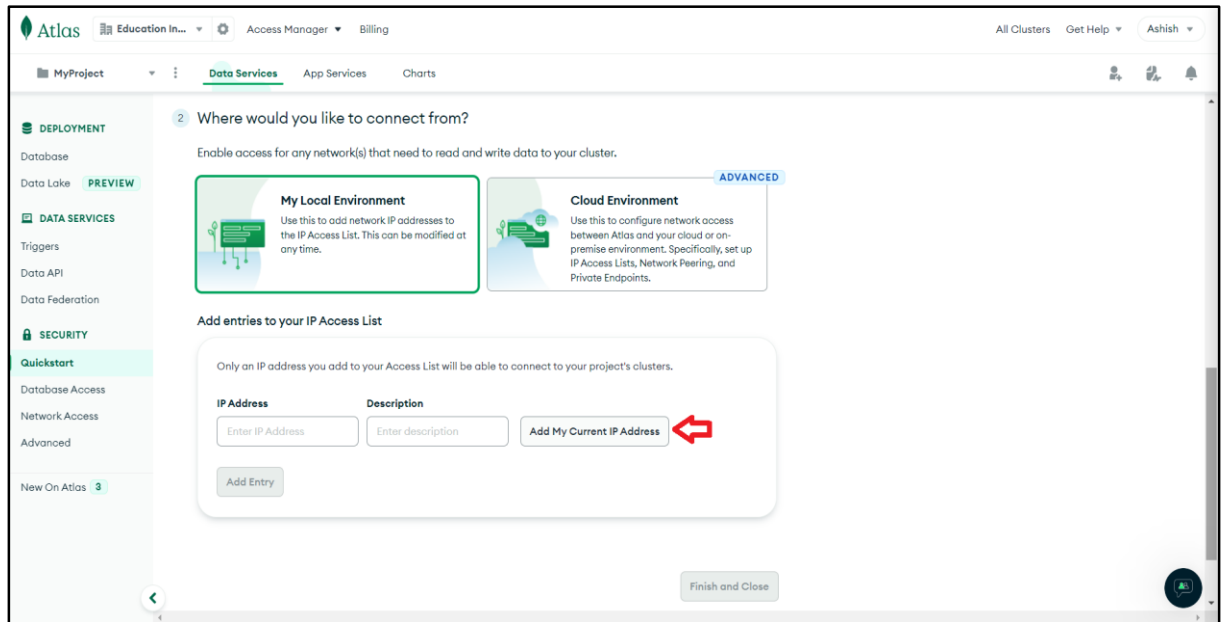
g) Choose the region to host your database and click on create cluster



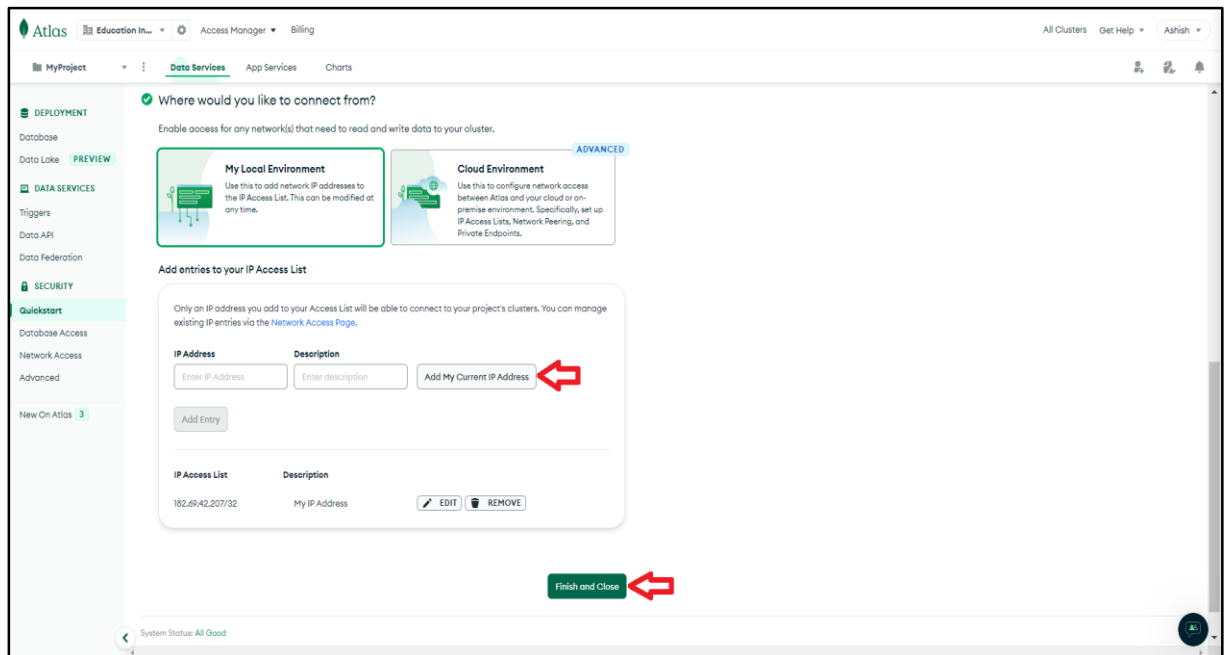
h) Set username and password. Note down the password as it will be used in next steps



- i) We will connect MongoDB using local environment. You need to add IP Address of the machine in order to connect to MongoDB cluster

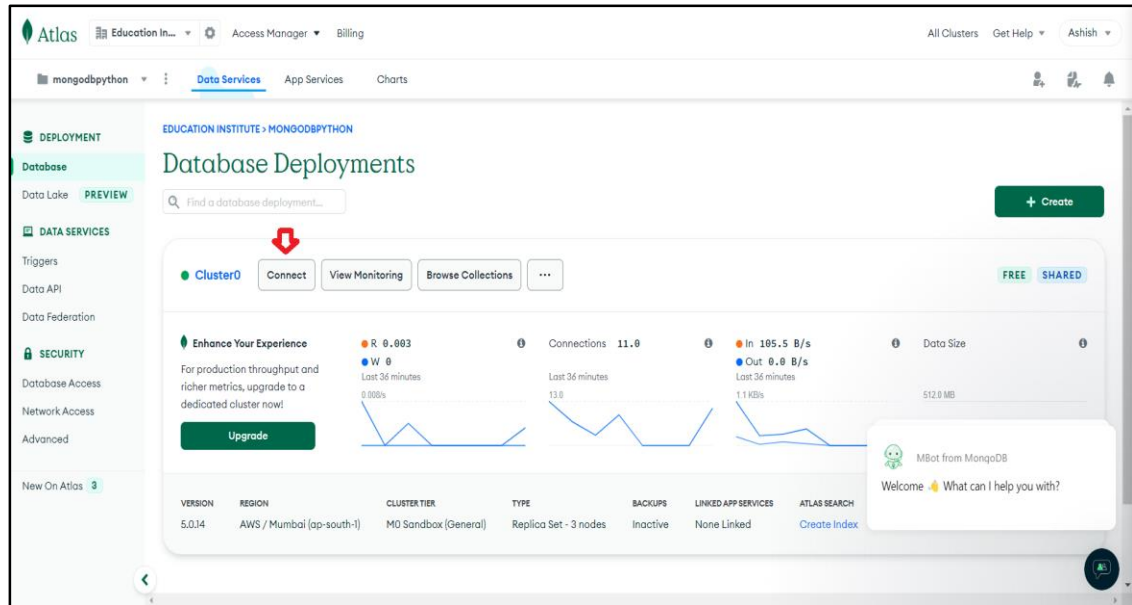


- j) Click on finish and close.

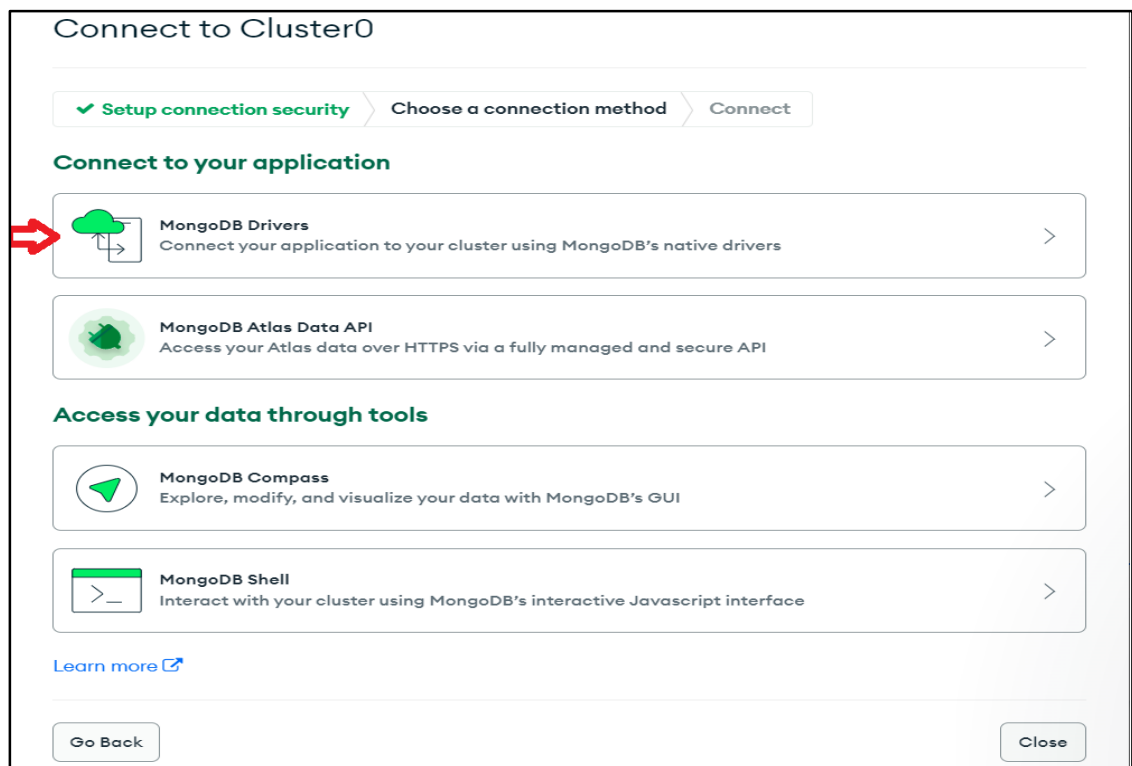


It will take around 3-4 minutes to setup the cluster.

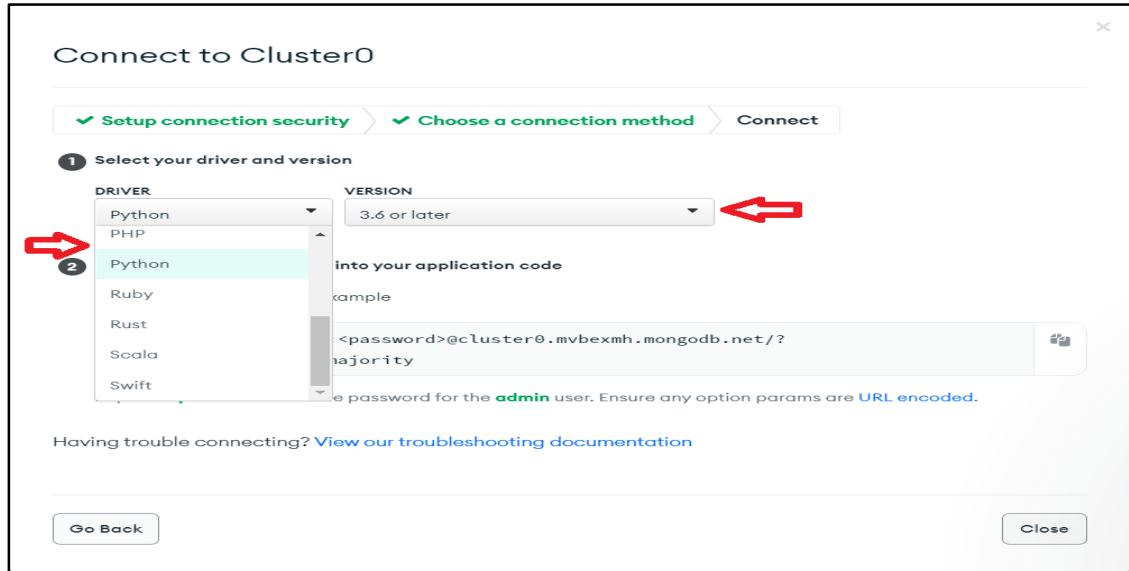
k) Choose the connect option as shown below



l) Further choose the first option under connect to your application using MongoDB drivers



m) Choose python under driver and appropriate version



Connect to Cluster0

✓ Setup connection security > ✓ Choose a connection method > Connect

1 Select your driver and version

DRIVER: Python, PHP, Ruby, Rust, Scala, Swift

VERSION: 3.6 or later

2 Add your connection string into your application code

Include full driver code example

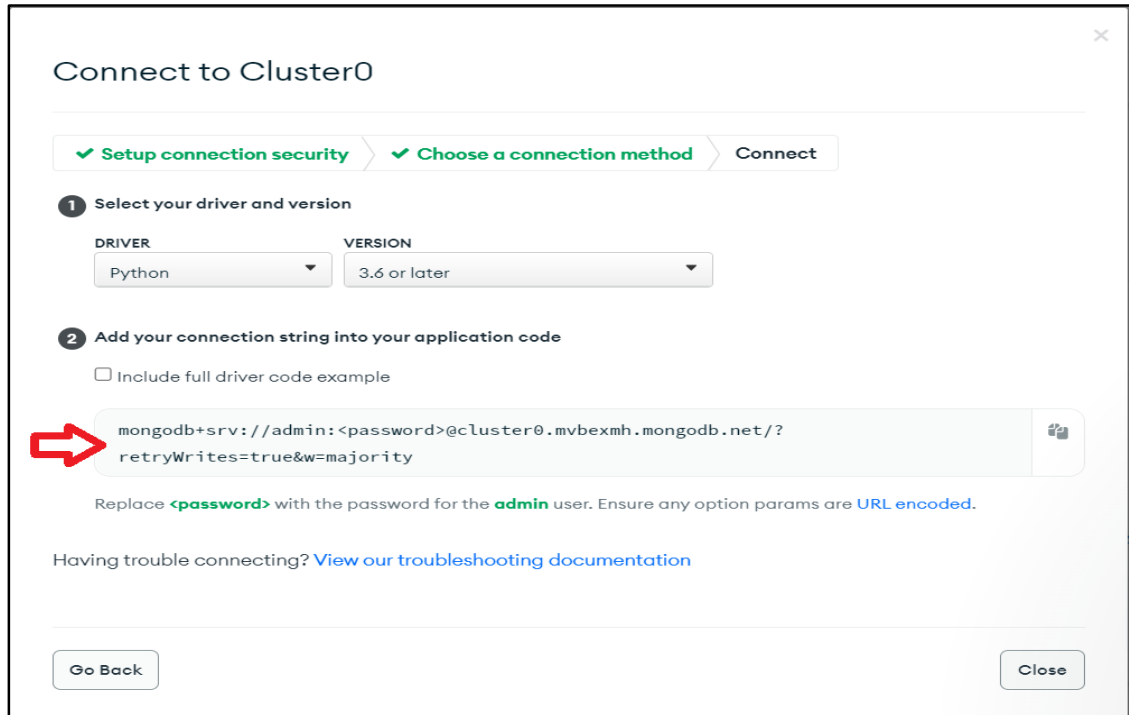
mongodb+srv://admin:<password>@cluster0.mvbexmh.mongodb.net/?retryWrites=true&w=majority

Replace <password> with the password for the admin user. Ensure any option params are URL encoded.

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back Close

n) Copy and save the below connection string. This will be used to connect to MongoDB using Python



Connect to Cluster0

✓ Setup connection security > ✓ Choose a connection method > Connect

1 Select your driver and version

DRIVER: Python

VERSION: 3.6 or later

2 Add your connection string into your application code

Include full driver code example

mongodb+srv://admin:<password>@cluster0.mvbexmh.mongodb.net/?retryWrites=true&w=majority

Replace <password> with the password for the admin user. Ensure any option params are URL encoded.

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back Close

- o) Now open the jupyter notebook in your machine and create a mongo client as shown below in the code. The connection string in previous step will be used to create a connection. In the connection string you have to replace <password> with actual password given in step h).**

```
In [2]: 1 #import pymongo package and create a MongoClient using the URL copied
        2 import pymongo
        3 client=pymongo.MongoClient('mongodb+srv://admin:admin@cluster0.mvbxmh.mongodb.net/?retryWrites=true&w=majority');
        4 print("Connection successful")

Connection successful
```

```
In [ ]: 1 |
```

- p) Once the connection is successful you can create a database as per the below code**

```
1 #Create a database test
2 db=client["test"]
```

This will create test database in MongoDB cluster

- q) You can create a collection as follows. This will create collection with name student.**

```
1 #Create a Collection Student
2 collection=db["student"]
```

- r) You can insert documents into collection as follows**

```
1 #Insert one record in collection
2 post={"fname":"ashish","name":"narang"}
3 collection.insert_one(post)
```

- s) You can insert multiple documents into collection as follows**

```
1 #Insert multiple records
2 collection.insert_many([{"fname":"rohan","lname":"narang"},
3                          {"fname":"rohit","lname":"kumar"},
4                          {"fname":"seema","lname":"dhawan"}])
```

```
{ '_id': ObjectId('6398e1a709af9f087f890c3a'), 'fname': 'ashish', 'name': 'narang' }
{ '_id': ObjectId('6398e21109af9f087f890c3b'), 'fname': 'rohan', 'lname': 'narang' }
{ '_id': ObjectId('6398e21109af9f087f890c3c'), 'fname': 'rohit', 'lname': 'kumar' }
{ '_id': ObjectId('6398e21109af9f087f890c3d'), 'fname': 'seema', 'lname': 'dhawan' }
```

t) Update a document

```
1 #update one record
2 myquery = { "fname": "rohan" }
3 newvalues = { "$set": { "lname": "goyal" } }
4 collection.update_one(myquery, newvalues)
```

u) Retrieve the updated document

```
1 #Reteriving the updated record
2 results_new=collection.find({"fname": "rohan"})
3 for result in results_new:
4     print(result)

{'_id': ObjectId('6398e21109af9f087f890c3b'), 'fname': 'rohan', 'lname': 'goyal'}
```

v) Delete a document

```
1 #Delete a document
2 myquery_del = { "fname": "rohit" }
3 collection.delete_one(myquery_del)
```

w) Print the collection and observe the deleted document not in the database.

```
1 results_del=collection.find({})
2 for result in results_del:
3     print(result)

{'_id': ObjectId('6398e1a709af9f087f890c3a'), 'fname': 'ashish', 'name': 'narang'}
{'_id': ObjectId('6398e21109af9f087f890c3b'), 'fname': 'rohan', 'lname': 'goyal'}
{'_id': ObjectId('6398e21109af9f087f890c3d'), 'fname': 'seema', 'lname': 'dhawan'}
```

```
1 results=collection.find({})
2 for result in results:
3     print(result)
```

4. Outputs/Results

- Students should be able to connect Mongo Atlas with python program
- Students should be able to execute MongoDB queries using python program

5. Observations

Students should carefully observe the steps to connect MongoDB Atlas with Python



References

- [MongoDB documentation](#)