MFLIX

Movies streaming app

Jyothish Kumar CHANDRASENAN GEETHAKUMARI

Msc Data Science and Analytics

Fall 2019- 2021

12-June-2020

Overview

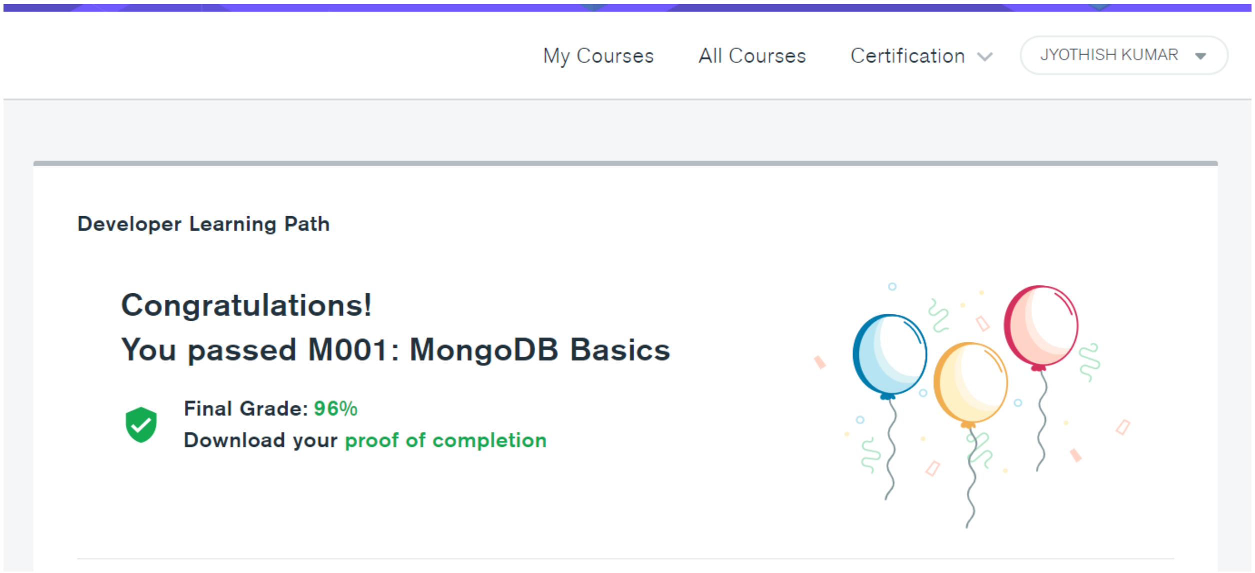
MFlix is a movie streaming application which uses MongoDB as the database in the back-end.

Mflix is composed of two main components:

- *Frontend*: All the UI functionality is already implemented which includes the built-in React application.
- *Backend*: The project that provides the necessary service to the application. The code flow is already implemented except some functions.

I have taken the below two courses in https://university.mongodb.com/ for learning MongoDB and implemented concepts in this project followed by the university.

- M001: MongoDB Basics

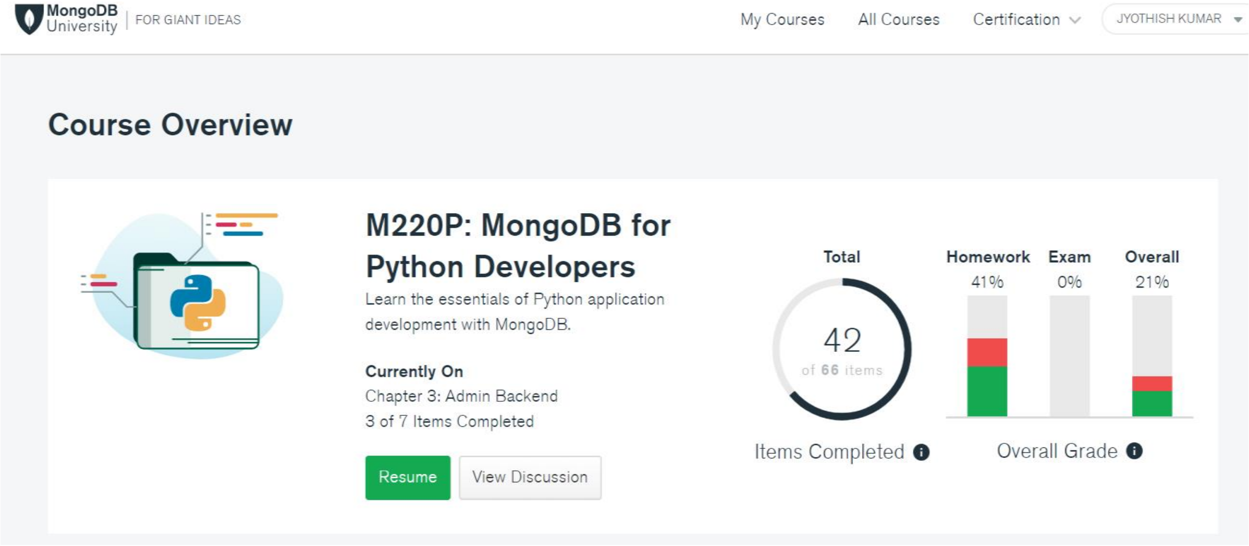


Developer Learning Path

Congratulations!
You passed M001: MongoDB Basics

Final Grade: 96%
Download your proof of completion

- MongoDB for Python Developers (In Progress)



Course Overview

M220P: MongoDB for Python Developers

Learn the essentials of Python application development with MongoDB.

Currently On
Chapter 3: Admin Backend
3 of 7 Items Completed

Total
42 of 66 items
Items Completed

Homework	Exam	Overall
41%	0%	21%

Overall Grade

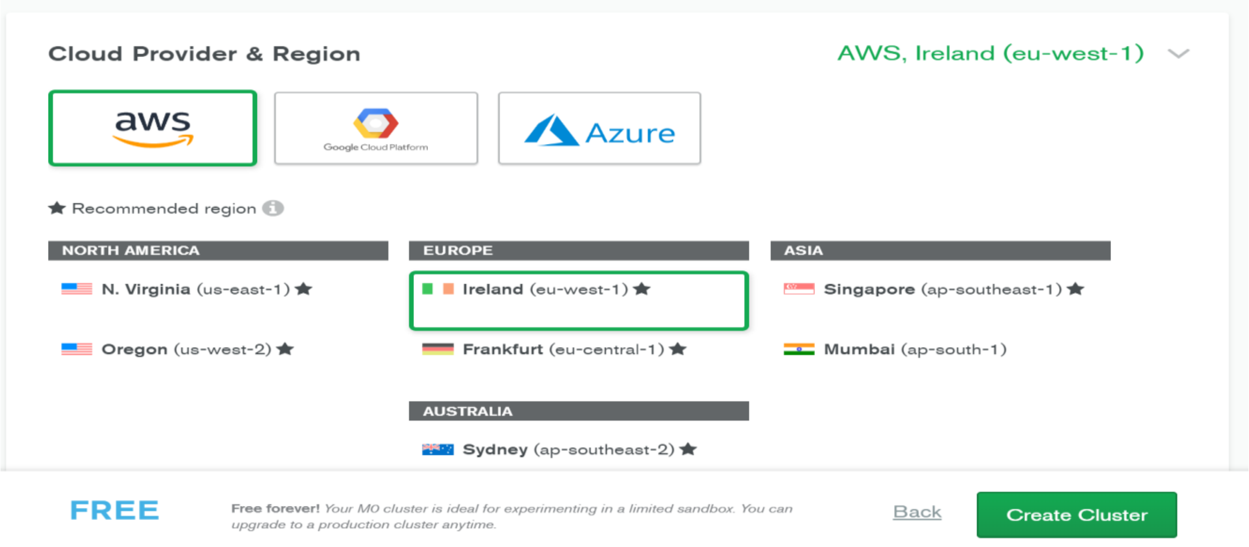
Code is uploaded to Github : https://github.com/jkcg-learning/MFlix

Tools Used

- MongoDB Compass – GUI interface for interacting with MongoDB
- Mongo shell – Shell mode for interacting with MongoDB
- MongoDB Atlas - Cloud Database
- Necessary Python Libraries (specified in the requirement.txt file)

Creation of Cluster in the MongoDB Atlas

- Create an Organization



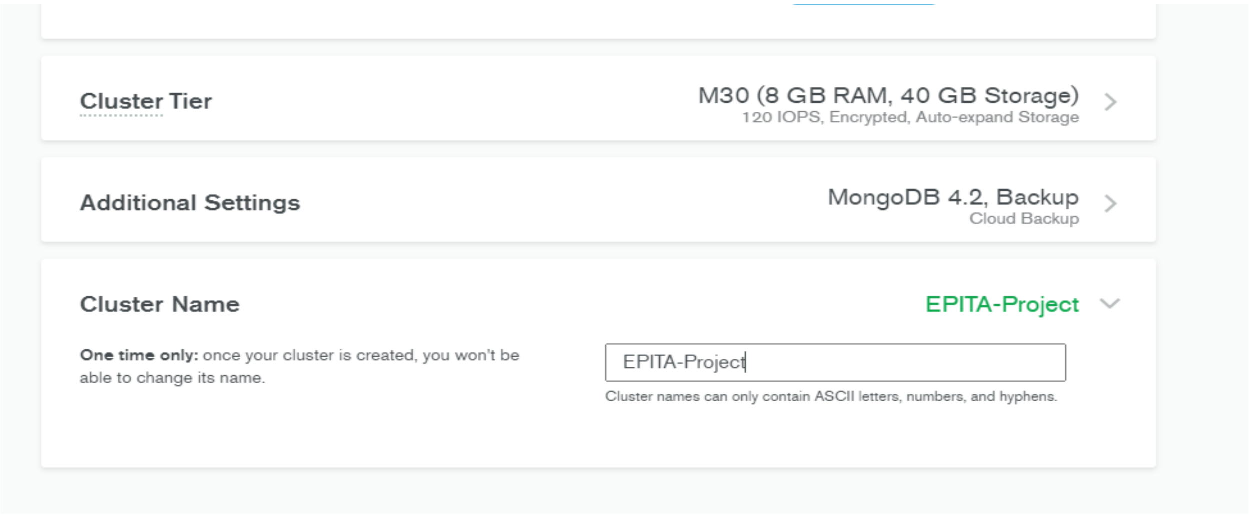
Cloud Provider & Region — AWS, Ireland (eu-west-1)

★ Recommended region

NORTH AMERICA	EUROPE	ASIA
N. Virginia (us-east-1) ★	Ireland (eu-west-1) ★	Singapore (ap-southeast-1) ★
Oregon (us-west-2) ★	Frankfurt (eu-central-1) ★	Mumbai (ap-south-1)
	AUSTRALIA	
	Sydney (ap-southeast-2) ★	

FREE

Free forever! *Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.*



Cluster Tier — M30 (8 GB RAM, 40 GB Storage)
120 IOPS, Encrypted, Auto-expand Storage

Additional Settings — MongoDB 4.2, Backup
Cloud Backup

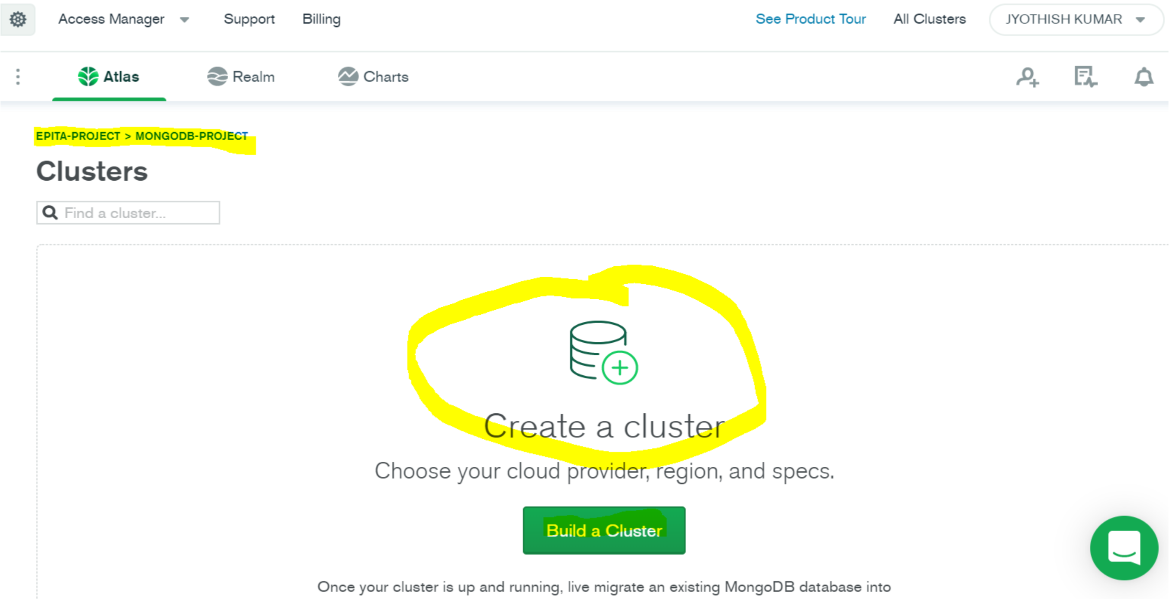
Cluster Name — EPITA-Project

One time only: once your cluster is created, you won't be able to change its name.

EPITA-Project

Cluster names can only contain ASCII letters, numbers, and hyphens.

- Create a Project & a Cluster



EPITA-PROJECT > MONGODB-PROJECT

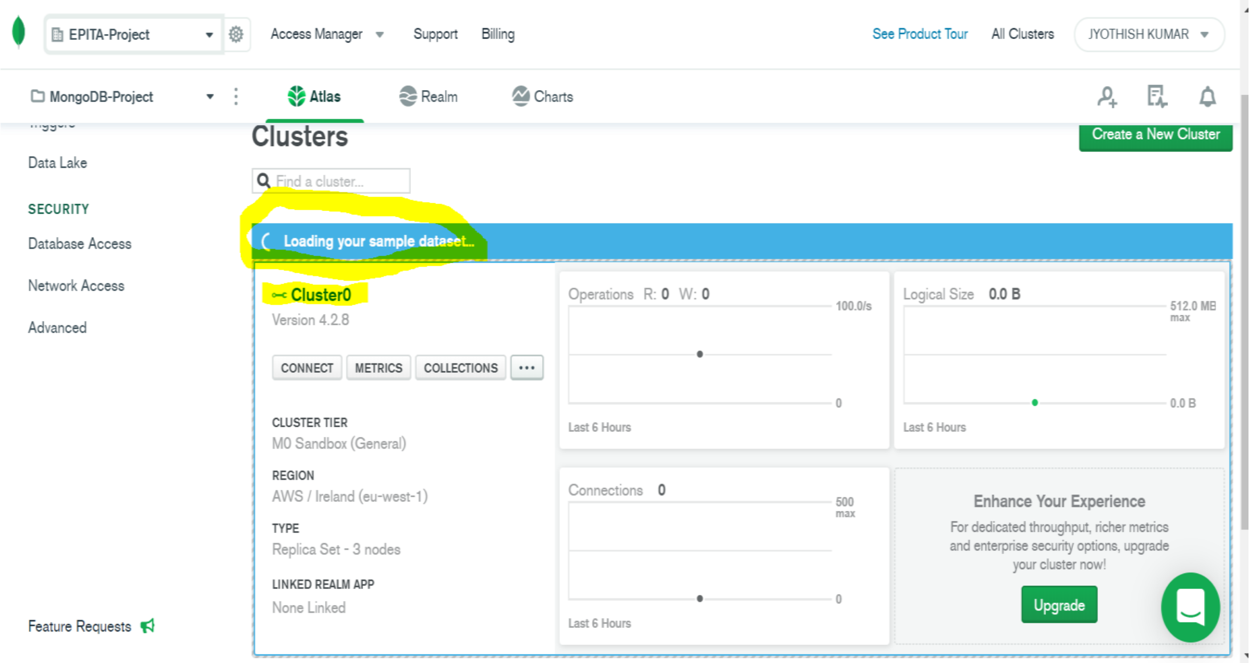
Clusters

Create a cluster

Choose your cloud provider, region, and specs.

Once your cluster is up and running, live migrate an existing MongoDB database into

By Default Mflix database is loaded into the cluster



Clusters

Loading your sample dataset...

Cluster0

Version 4.2.8

CLUSTER TIER
M0 Sandbox (General)

REGION
AWS / Ireland (eu-west-1)

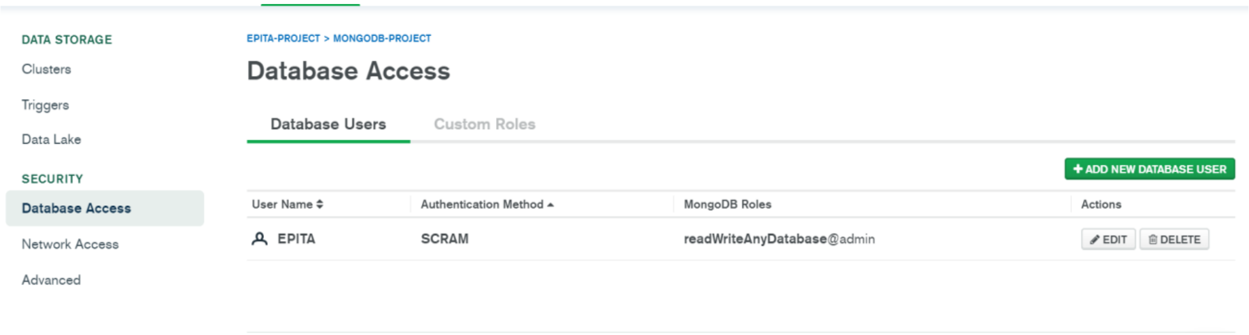
TYPE
Replica Set - 3 nodes

LINKED REALM APP
None Linked

Enhance Your Experience

For dedicated throughput, richer metrics and enterprise security options, upgrade your cluster now!

- Provide Database Access and Network Access for the Cluster



EPITA-PROJECT > MONGODB-PROJECT

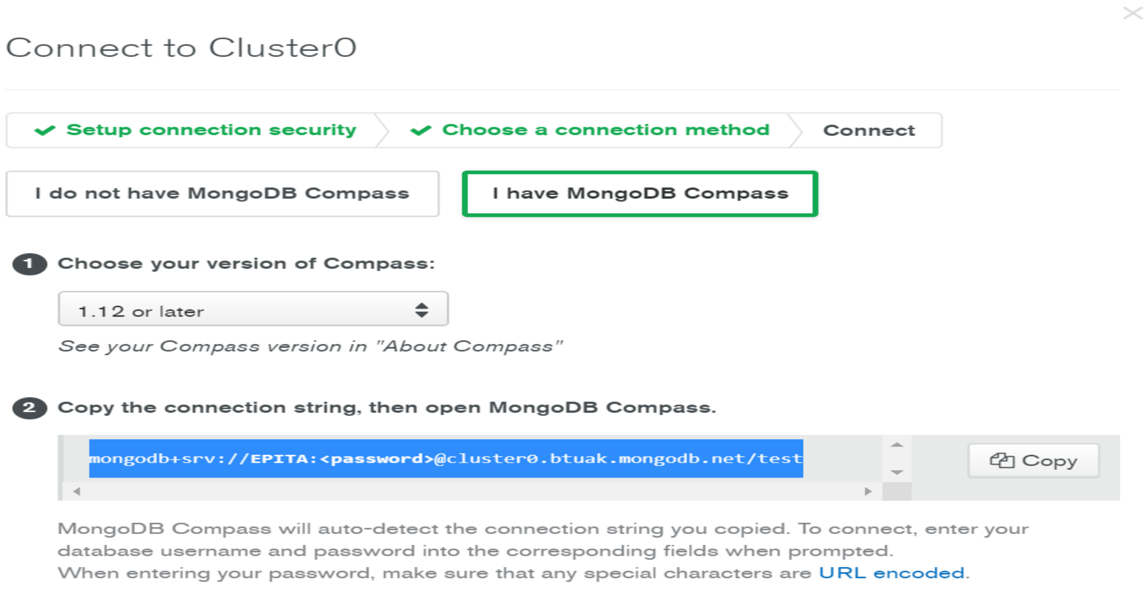
Database Access

User Name ⇕	Authentication Method ▴	MongoDB Roles	Actions
EPITA	SCRAM	readWriteAnyDatabase@admin	✎ EDIT 🗑 DELETE

Connect cluster using Mongo shell and MongoDB Compass

Mongo Shell

```
mongo "mongodb+srv://cluster0.btuak.mongodb.net/" --username EPITA
```


✓ Setup connection security
✓ Choose a connection method
Connect
I do not have MongoDB Compass
I have MongoDB Compass
Copy

Connect to Cluster0

1 Choose your version of Compass:

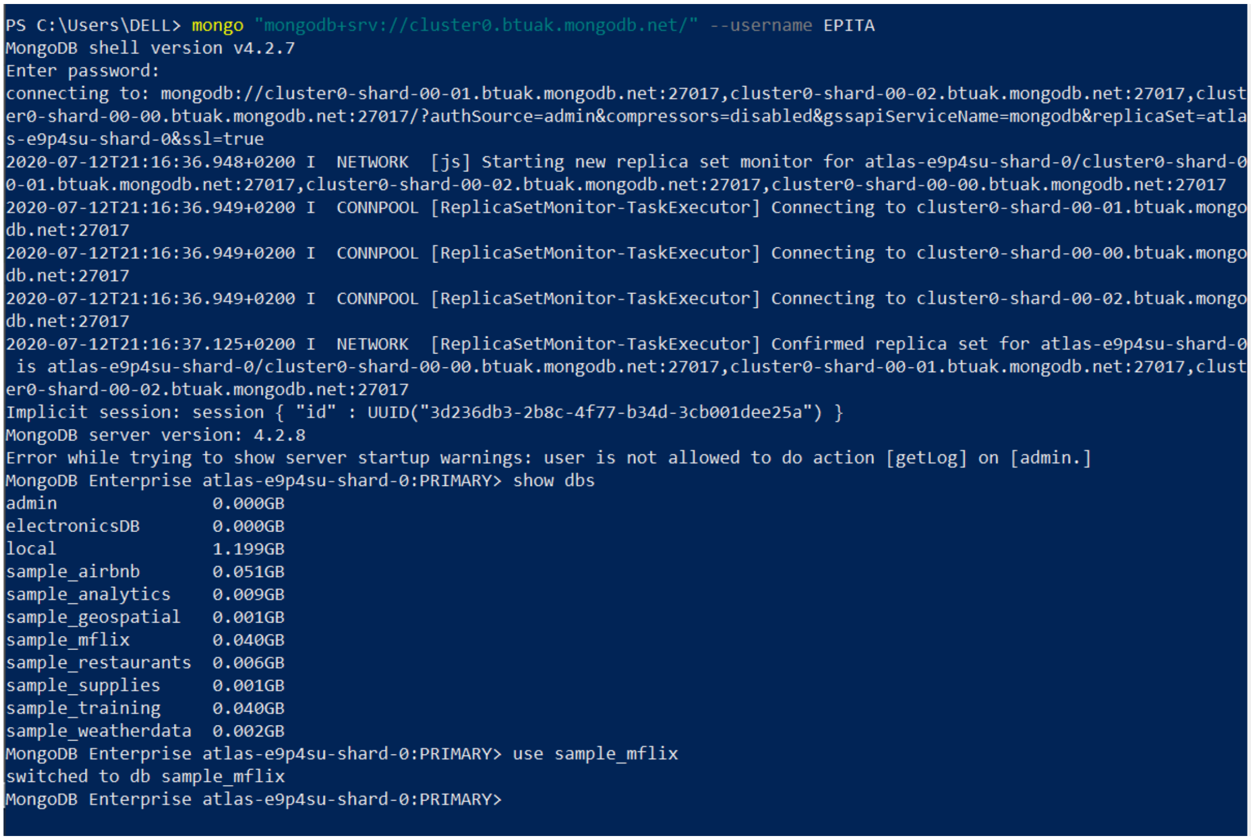
1.12 or later

See your Compass version in "About Compass"

2 Copy the connection string, then open MongoDB Compass.

```
mongodb+srv://EPITA:<password>@cluster0.btuak.mongodb.net/test
```

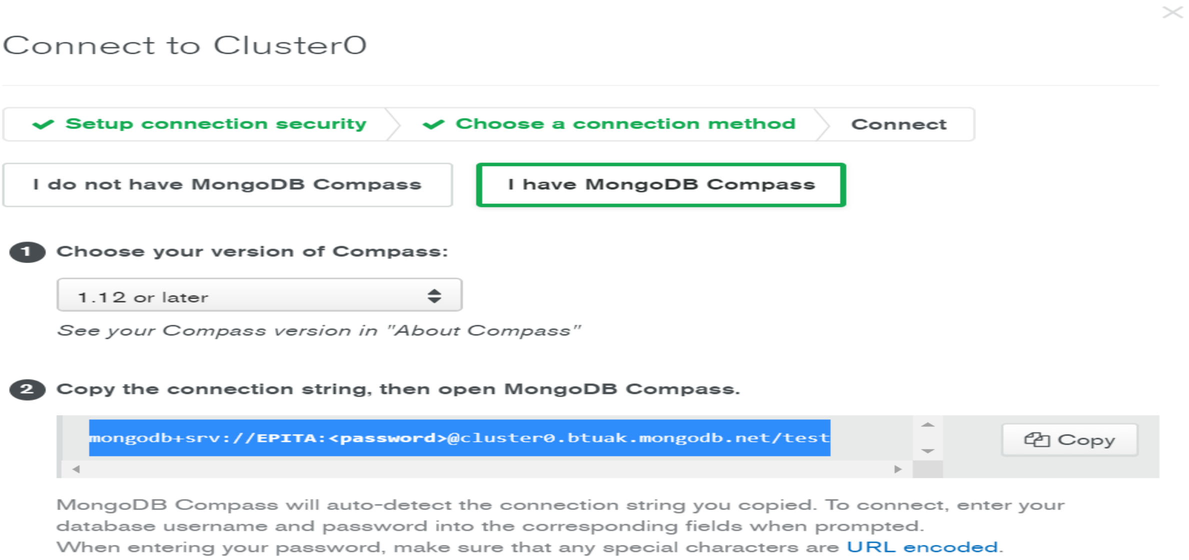
MongoDB Compass will auto-detect the connection string you copied. To connect, enter your database username and password into the corresponding fields when prompted.
When entering your password, make sure that any special characters are URL encoded.



```
PS C:\Users\DELL> mongo "mongodb+srv://cluster0.btuak.mongodb.net/" --username EPITA
MongoDB shell version v4.2.7
Enter password:
connecting to: mongodb://cluster0-shard-00-01.btuak.mongodb.net:27017,cluster0-shard-00-02.btuak.mongodb.net:27017,clust
er0-shard-00-00.btuak.mongodb.net:27017/?authSource=admin&compressors=disabled&gssapiServiceName=mongodb&replicaSet=atla
s-e9p4su-shard-0&ssl=true
2020-07-12T21:16:36.948+0200 I  NETWORK  [js] Starting new replica set monitor for atlas-e9p4su-shard-0/cluster0-shard-0
0-01.btuak.mongodb.net:27017,cluster0-shard-00-02.btuak.mongodb.net:27017,cluster0-shard-00-00.btuak.mongodb.net:27017
2020-07-12T21:16:36.949+0200 I  CONNPOOL [ReplicaSetMonitor-TaskExecutor] Connecting to cluster0-shard-00-01.btuak.mongo
db.net:27017
2020-07-12T21:16:36.949+0200 I  CONNPOOL [ReplicaSetMonitor-TaskExecutor] Connecting to cluster0-shard-00-00.btuak.mongo
db.net:27017
2020-07-12T21:16:36.949+0200 I  CONNPOOL [ReplicaSetMonitor-TaskExecutor] Connecting to cluster0-shard-00-02.btuak.mongo
db.net:27017
2020-07-12T21:16:37.125+0200 I  NETWORK  [ReplicaSetMonitor-TaskExecutor] Confirmed replica set for atlas-e9p4su-shard-0
 is atlas-e9p4su-shard-0/cluster0-shard-00-00.btuak.mongodb.net:27017,cluster0-shard-00-01.btuak.mongodb.net:27017,clust
er0-shard-00-02.btuak.mongodb.net:27017
Implicit session: session { "id" : UUID("3d236db3-2b8c-4f77-b34d-3cb001dee25a") }
MongoDB server version: 4.2.8
Error while trying to show server startup warnings: user is not allowed to do action [getLog] on [admin.]
MongoDB Enterprise atlas-e9p4su-shard-0:PRIMARY> show dbs
admin                0.000GB
electronicsDB        0.000GB
local                1.199GB
sample_airbnb        0.051GB
sample_analytics     0.009GB
sample_geospatial    0.001GB
sample_mflix         0.040GB
sample_restaurants   0.006GB
sample_supplies      0.001GB
sample_training      0.040GB
sample_weatherdata   0.002GB
MongoDB Enterprise atlas-e9p4su-shard-0:PRIMARY> use sample_mflix
switched to db sample_mflix
MongoDB Enterprise atlas-e9p4su-shard-0:PRIMARY>
```

MongoDB Compass

mongodb+srv://EPITA:<password>@cluster0.btuak.mongodb.net/



Connect to Cluster0

✓ Setup connection security ✓ Choose a connection method Connect

I do not have MongoDB Compass | I have MongoDB Compass

1 Choose your version of Compass:

1.12 or later

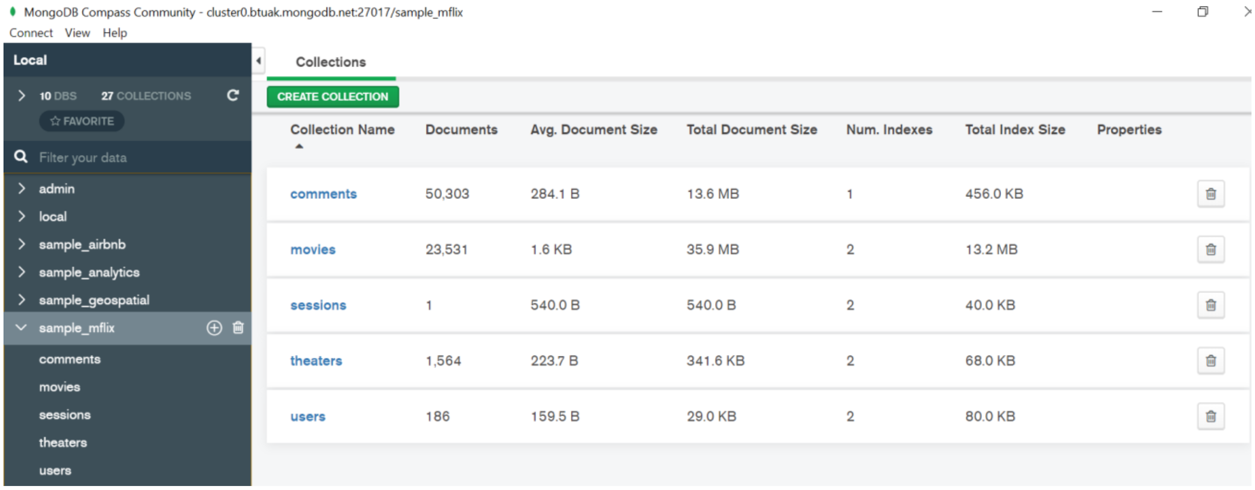
See your Compass version in "About Compass"

2 Copy the connection string, then open MongoDB Compass.

```
mongodb+srv://EPITA:<password>@cluster0.btuak.mongodb.net/test
```

Copy

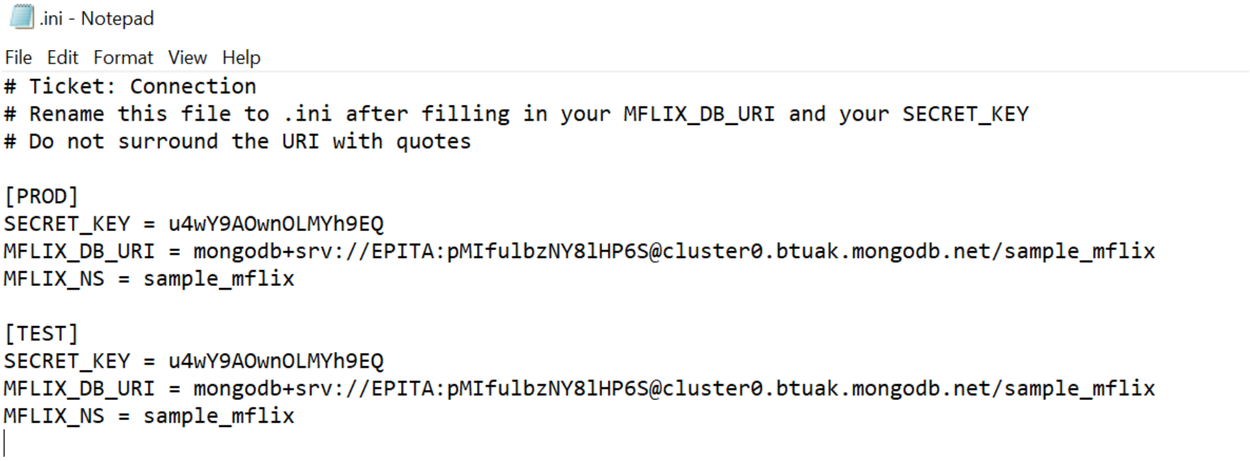
MongoDB Compass will auto-detect the connection string you copied. To connect, enter your database username and password into the corresponding fields when prompted.
When entering your password, make sure that any special characters are URL encoded.



Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
comments	50,303	284.1 B	13.6 MB	1	456.0 KB	
movies	23,531	1.6 KB	35.9 MB	2	13.2 MB	
sessions	1	540.0 B	540.0 B	2	40.0 KB	
theaters	1,564	223.7 B	341.6 KB	2	68.0 KB	
users	186	159.5 B	29.0 KB	2	80.0 KB	

Launch the MFlix Application

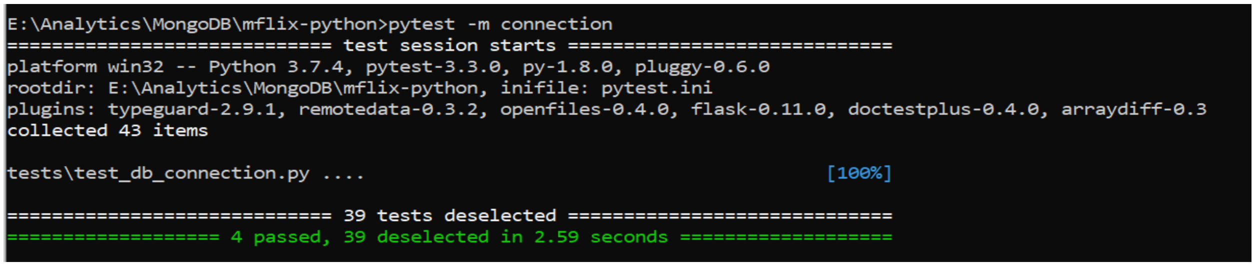
- Download the project from the university
- Make the necessary changes in the .ini file which will connect the application to the database in our cluster.



```
# Ticket: Connection
# Rename this file to .ini after filling in your MFLIX_DB_URI and your SECRET_KEY
# Do not surround the URI with quotes

[PROD]
SECRET_KEY = u4wY9AOwnOLMYh9EQ
MFLIX_DB_URI = mongodb+srv://EPITA:pMIfulbzNY8lHP6S@cluster0.btuak.mongodb.net/sample_mflix
MFLIX_NS = sample_mflix

[TEST]
SECRET_KEY = u4wY9AOwnOLMYh9EQ
MFLIX_DB_URI = mongodb+srv://EPITA:pMIfulbzNY8lHP6S@cluster0.btuak.mongodb.net/sample_mflix
MFLIX_NS = sample_mflix
```

- Test the connection

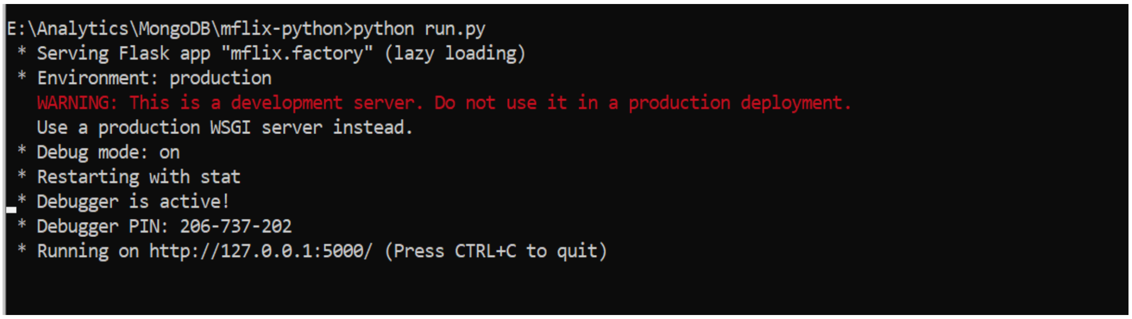
```
pytest -m connection
```



```
E:\Analytics\MongoDB\mflix-python>pytest -m connection
============================= test session starts =============================
platform win32 -- Python 3.7.4, pytest-3.3.0, py-1.8.0, pluggy-0.6.0
rootdir: E:\Analytics\MongoDB\mflix-python, inifile: pytest.ini
plugins: typeguard-2.9.1, remotedata-0.3.2, openfiles-0.4.0, flask-0.11.0, doctestplus-0.4.0, arraydiff-0.3
collected 43 items

tests\test_db_connection.py ....                                        [100%]

============================= 39 tests deselected =============================
=================== 4 passed, 39 deselected in 2.59 seconds ===================
```

- **Run the application**

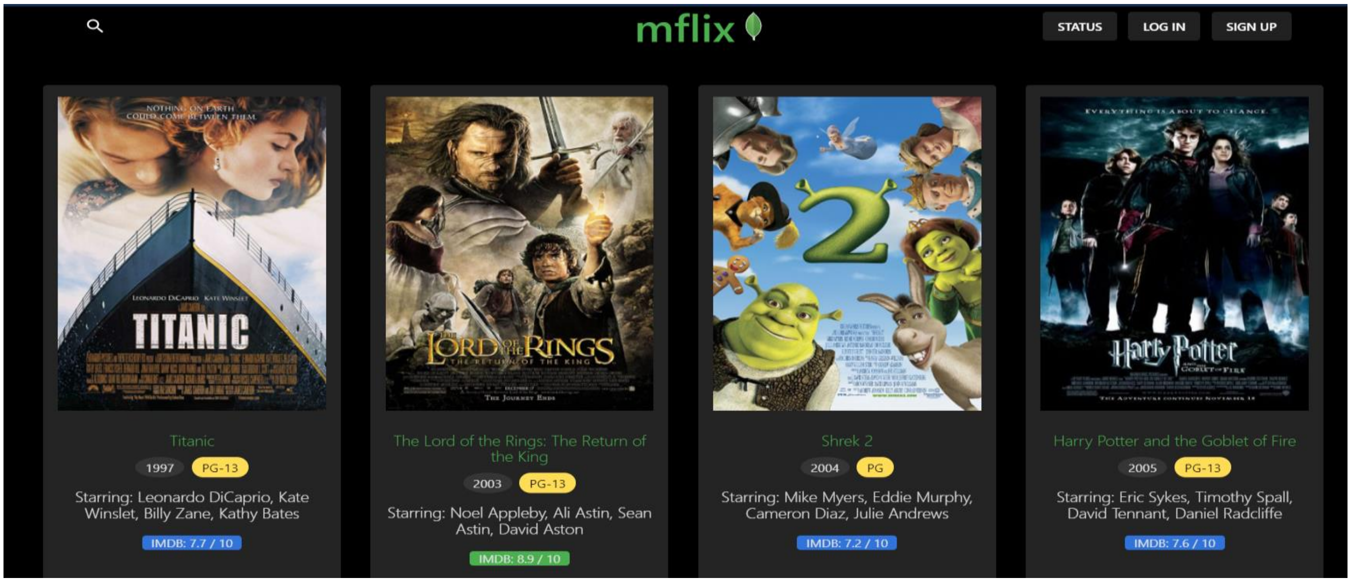
  python run.py



```
E:\Analytics\MongoDB\mflix-python>python run.py
 * Serving Flask app "mflix.factory" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 206-737-202
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Application is launched and can be accessed using

http://127.0.0.1:5000/



Titanic
1997 PG-13
Starring: Leonardo DiCaprio, Kate Winslet, Billy Zane, Kathy Bates
IMDB: 7.7 / 10

The Lord of the Rings: The Return of the King
2003 PG-13
Starring: Noel Appleby, Ali Astin, Sean Astin, David Aston
IMDB: 8.9 / 10

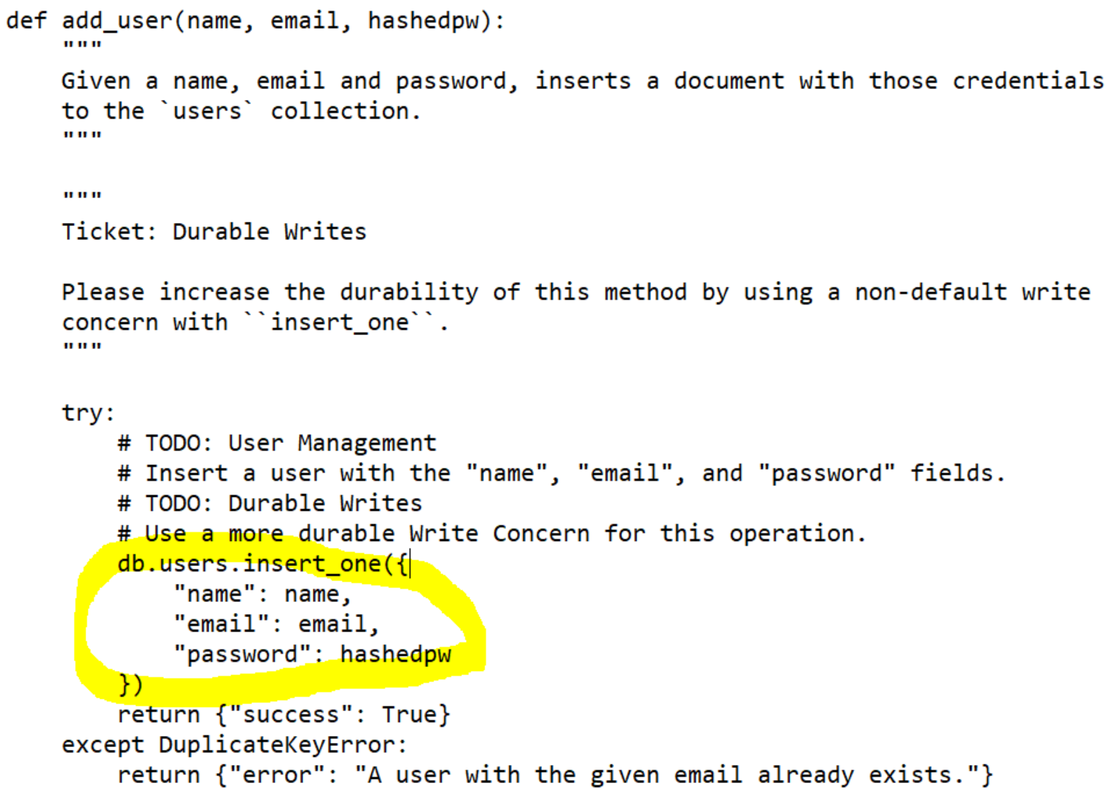
Shrek 2
2004 PG
Starring: Mike Myers, Eddie Murphy, Cameron Diaz, Julie Andrews
IMDB: 7.2 / 10

Harry Potter and the Goblet of Fire
2005 PG-13
Starring: Eric Sykes, Timothy Spall, David Tennant, Daniel Radcliffe
IMDB: 7.6 / 10

CRUD Operations

• CREATE

A new user can be created using the front-end and back-end function is written to insert it into MongoDB



```
def add_user(name, email, hashedpw):
    """
    Given a name, email and password, inserts a document with those credentials
    to the `users` collection.
    """

    """
    Ticket: Durable Writes

    Please increase the durability of this method by using a non-default write
    concern with ``insert_one``.
    """

    try:
        # TODO: User Management
        # Insert a user with the "name", "email", and "password" fields.
        # TODO: Durable Writes
        # Use a more durable Write Concern for this operation.
        db.users.insert_one({
            "name": name,
            "email": email,
            "password": hashedpw
        })
        return {"success": True}
    except DuplicateKeyError:
        return {"error": "A user with the given email already exists."}
```

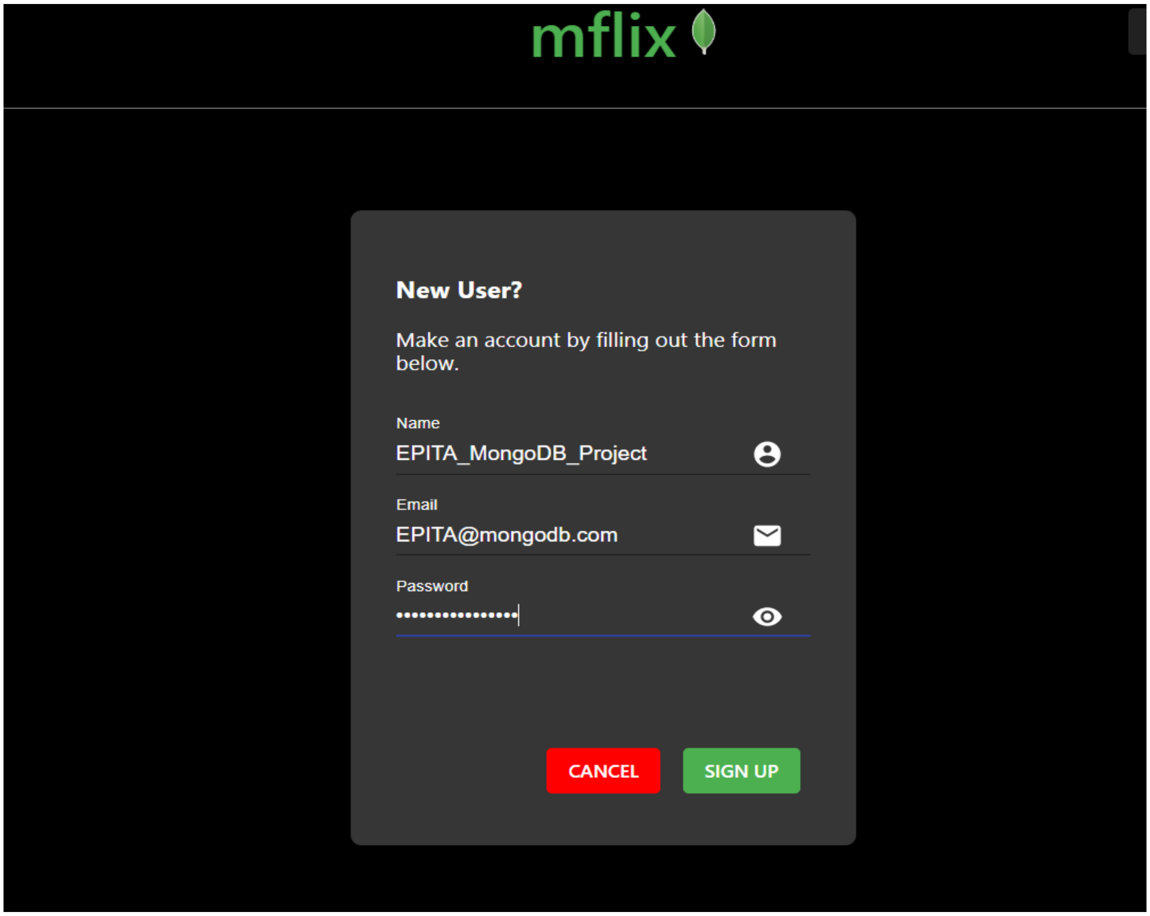
Using collections.insert_one() from MongoDB

db.users.insert_one({

"name": name,

"email": email,

"password": hashedpw})



New User?

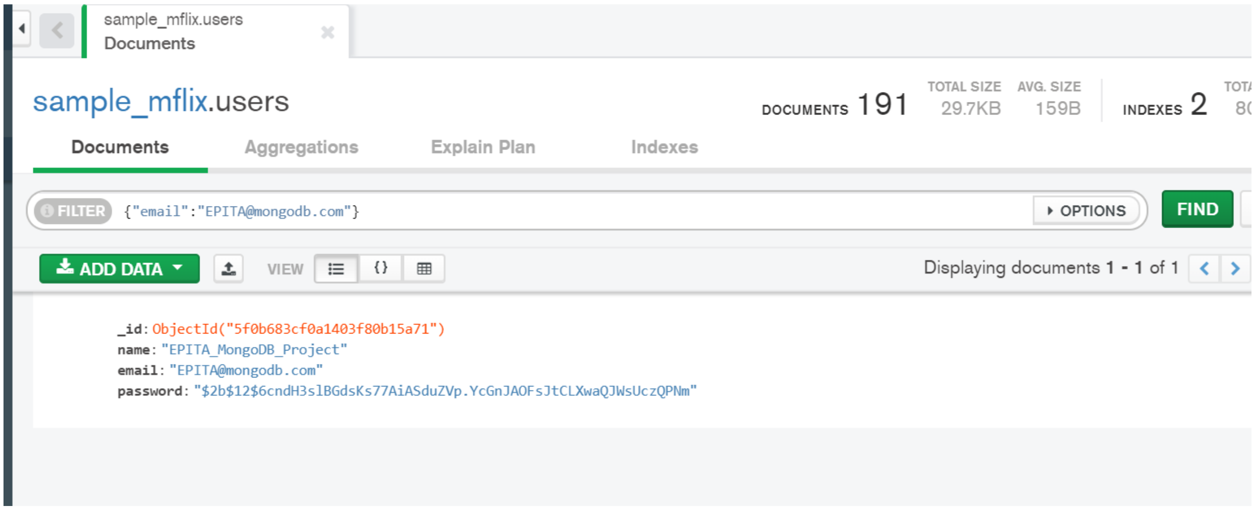
Make an account by filling out the form below.

Name
EPITA_MongoDB_Project

Email
EPITA@mongodb.com

Password
••••••••••••••••

The details are inserted in the users collection of sample_mflix database

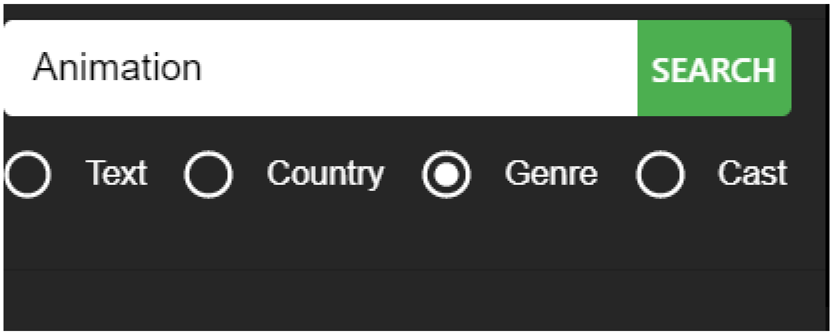


```
{"email":"EPITA@mongodb.com"}
```

```
_id: ObjectId("5f0b683cf0a1403f80b15a71")
name: "EPITA_MongoDB_Project"
email: "EPITA@mongodb.com"
password: "$2b$12$6cndH3slBGdsKs77AiASduZVp.YcGnJAOFsJtCLXwaQJWsUczQPNm"
```

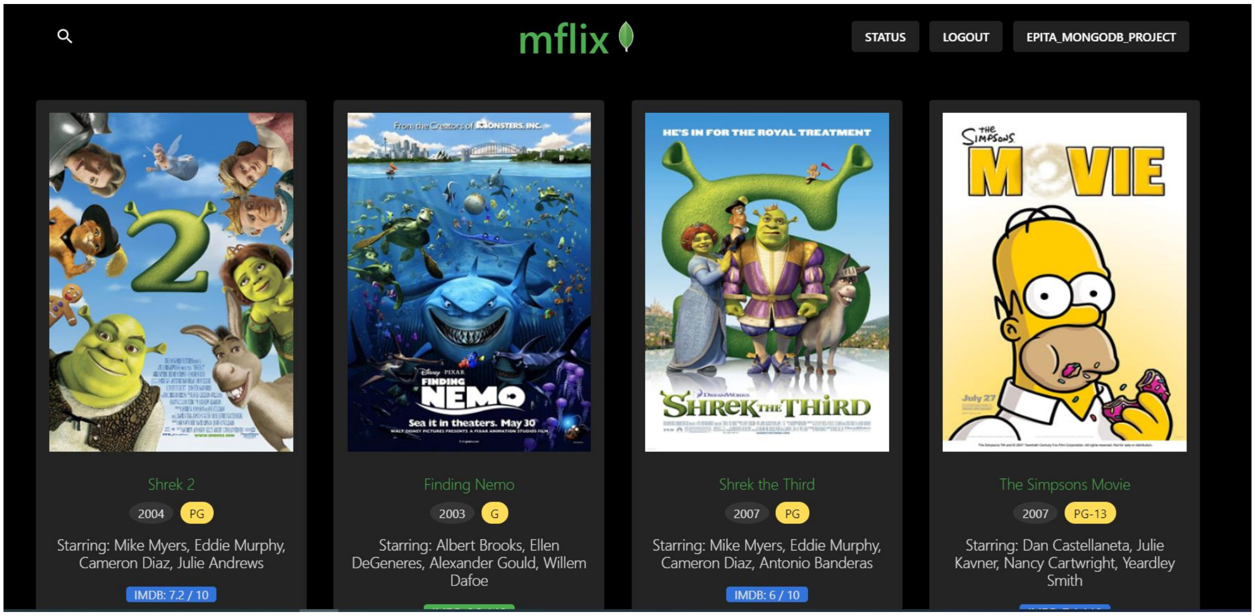
- **Read**

Search for movies based on Genres



Animation

○ Text ○ Country ● Genre ○ Cast



Shrek 2
2004 PG
Starring: Mike Myers, Eddie Murphy, Cameron Diaz, Julie Andrews
IMDB: 7.2 / 10

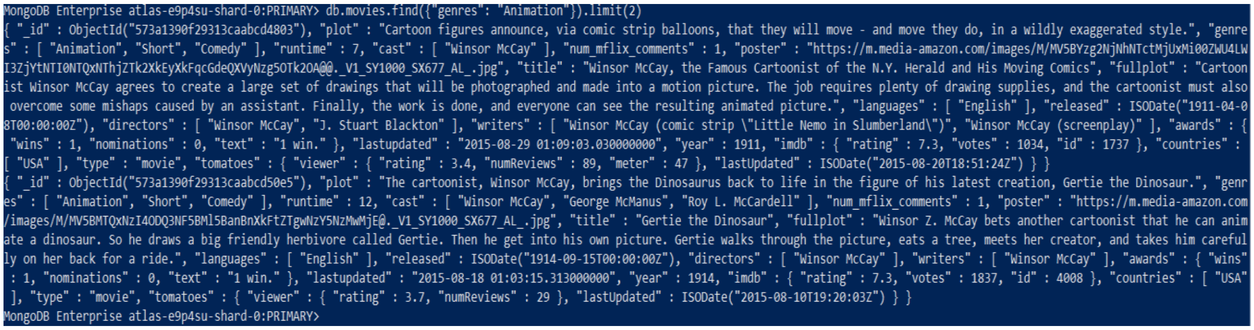
Finding Nemo
2003 G
Starring: Albert Brooks, Ellen DeGeneres, Alexander Gould, Willem Dafoe

Shrek the Third
2007 PG
Starring: Mike Myers, Eddie Murphy, Cameron Diaz, Antonio Banderas
IMDB: 6 / 10

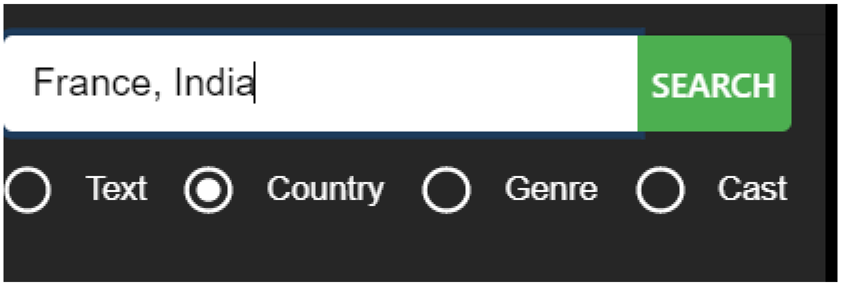
The Simpsons Movie
2007 PG-13
Starring: Dan Castellaneta, Julie Kavner, Nancy Cartwright, Yeardley Smith

Using collections.find() from MongoDB

```
db.movies.find({"genres": "Animation"}).limit(2)
```

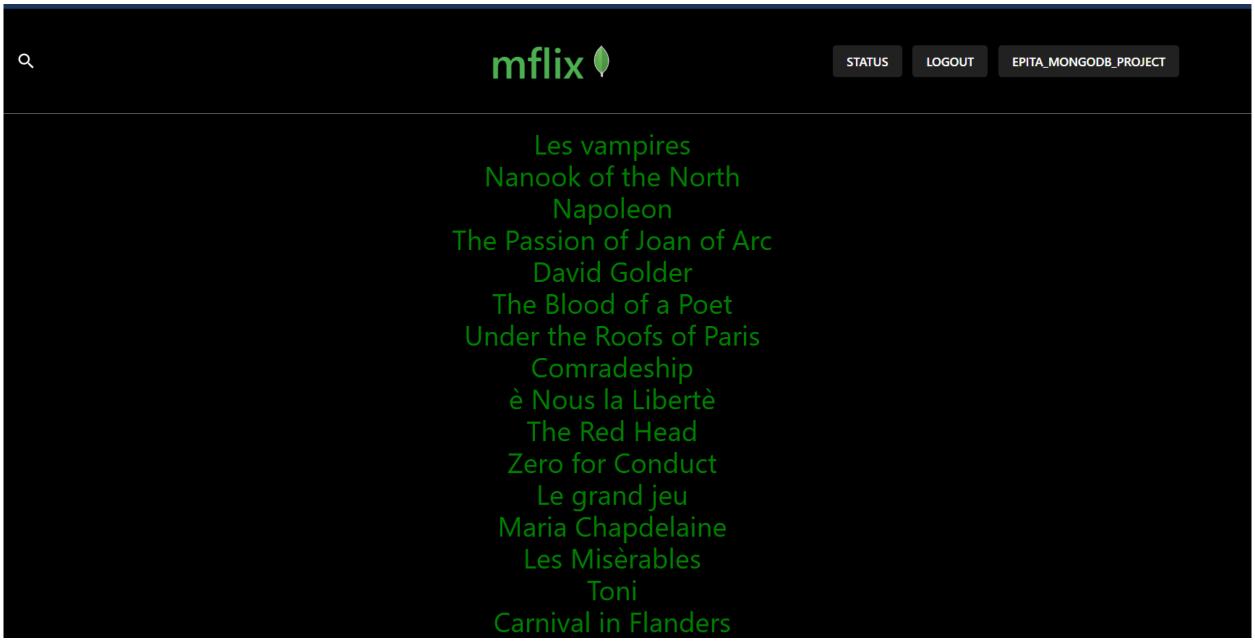


```
MongoDB Enterprise atlas-e9p4su-shard-0:PRIMARY> db.movies.find({"genres": "Animation"}).limit(2)
{ "_id" : ObjectId("573a1390f29313caabcd4803"), "plot" : "Cartoon figures announce, via comic strip balloons, that they will move - and move they do, in a wildly exaggerated style.", "genre
s" : [ "Animation", "Short", "Comedy" ], "runtime" : 7, "cast" : [ "Winsor McCay" ], "num_mflix_comments" : 1, "poster" : "https://m.media-amazon.com/images/M/MV5BYzg2NjNhNTctMjUxMi00ZWU4LW
I3ZjYtNTI0NTQxNThjZTk2XkEyXkFqcGdeQXVyNzg5OTk2OA@@._V1_SY1000_SX677_AL_.jpg", "title" : "Winsor McCay, the Famous Cartoonist of the N.Y. Herald and His Moving Comics", "fullplot" : "Cartoon
ist Winsor McCay agrees to create a large set of drawings that will be photographed and made into a motion picture. The job requires plenty of drawing supplies, and the cartoonist must also
 overcome some mishaps caused by an assistant. Finally, the work is done, and everyone can see the resulting animated picture.", "languages" : [ "English" ], "released" : ISODate("1911-04-0
8T00:00:00Z"), "directors" : [ "Winsor McCay", "J. Stuart Blackton" ], "writers" : [ "Winsor McCay (comic strip \"Little Nemo in Slumberland\")", "Winsor McCay (screenplay)" ], "awards" : {
 "wins" : 1, "nominations" : 0, "text" : "1 win." }, "lastupdated" : "2015-08-29 01:09:03.030000000", "year" : 1911, "imdb" : { "rating" : 7.3, "votes" : 1034, "id" : 1737 }, "countries" :
[ "USA" ], "type" : "movie", "tomatoes" : { "viewer" : { "rating" : 3.4, "numReviews" : 89, "meter" : 47 }, "lastUpdated" : ISODate("2015-08-20T18:51:24Z") } }
{ "_id" : ObjectId("573a1390f29313caabcd50e5"), "plot" : "The cartoonist, Winsor McCay, brings the Dinosaurus back to life in the figure of his latest creation, Gertie the Dinosaur.", "genr
es" : [ "Animation", "Short", "Comedy" ], "runtime" : 12, "cast" : [ "Winsor McCay", "George McManus", "Roy L. McCardell" ], "num_mflix_comments" : 1, "poster" : "https://m.media-amazon.com
/images/M/MV5BMTQxNzI4ODQ3NF5BMl5BanBnXkFtZTgwNzY5NzMwMjE@._V1_SY1000_SX677_AL_.jpg", "title" : "Gertie the Dinosaur", "fullplot" : "Winsor Z. McCay bets another cartoonist that he can anim
ate a dinosaur. So he draws a big friendly herbivore called Gertie. Then he get into his own picture. Gertie walks through the picture, eats a tree, meets her creator, and takes him careful
ly on her back for a ride.", "languages" : [ "English" ], "released" : ISODate("1914-09-15T00:00:00Z"), "directors" : [ "Winsor McCay" ], "writers" : [ "Winsor McCay" ], "awards" : { "wins"
 : 1, "nominations" : 0, "text" : "1 win." }, "lastupdated" : "2015-08-18 01:03:15.313000000", "year" : 1914, "imdb" : { "rating" : 7.3, "votes" : 1837, "id" : 4008 }, "countries" : [ "USA"
 ], "type" : "movie", "tomatoes" : { "viewer" : { "rating" : 3.7, "numReviews" : 29 }, "lastUpdated" : ISODate("2015-08-10T19:20:03Z") } }
MongoDB Enterprise atlas-e9p4su-shard-0:PRIMARY>
```



France, India

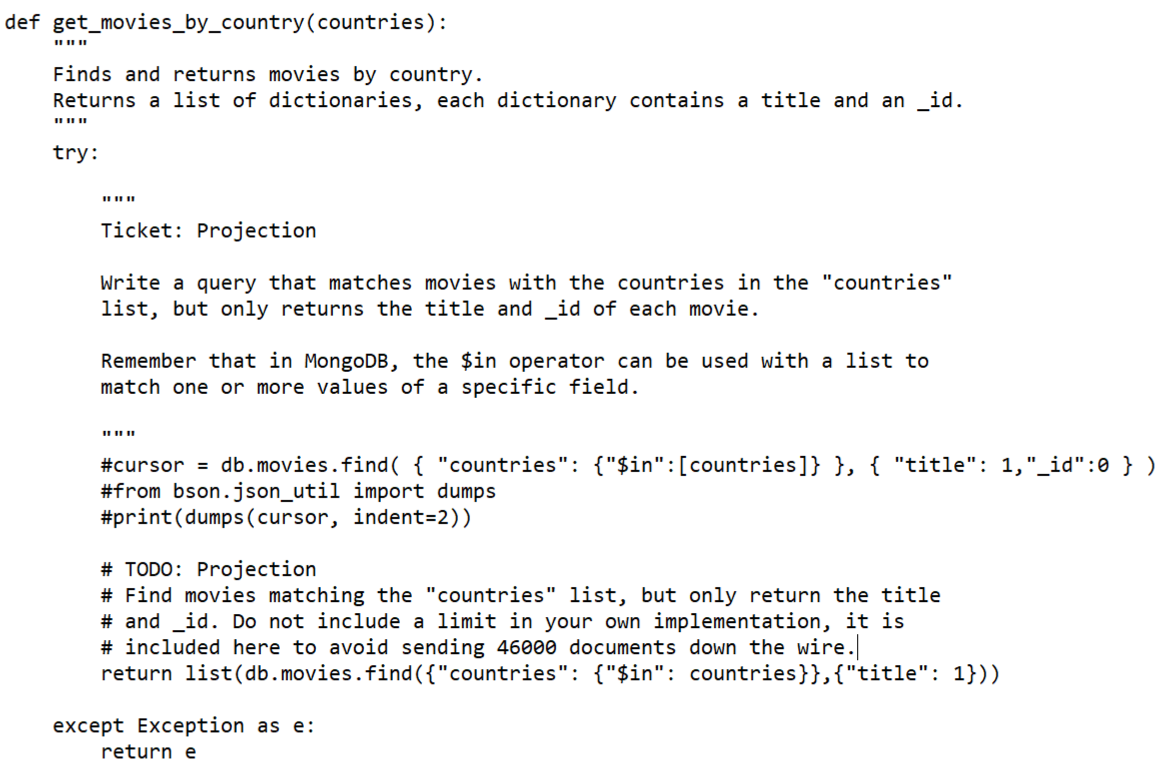
○ Text ● Country ○ Genre ○ Cast



Les vampires
Nanook of the North
Napoleon
The Passion of Joan of Arc
David Golder
The Blood of a Poet
Under the Roofs of Paris
Comradeship
è Nous la Libertè
The Red Head
Zero for Conduct
Le grand jeu
Maria Chapdelaine
Les Misèrables
Toni
Carnival in Flanders

db.movies.find({ "countries": {"$in":[countries]} }, { "title": 1,"_id":0 })

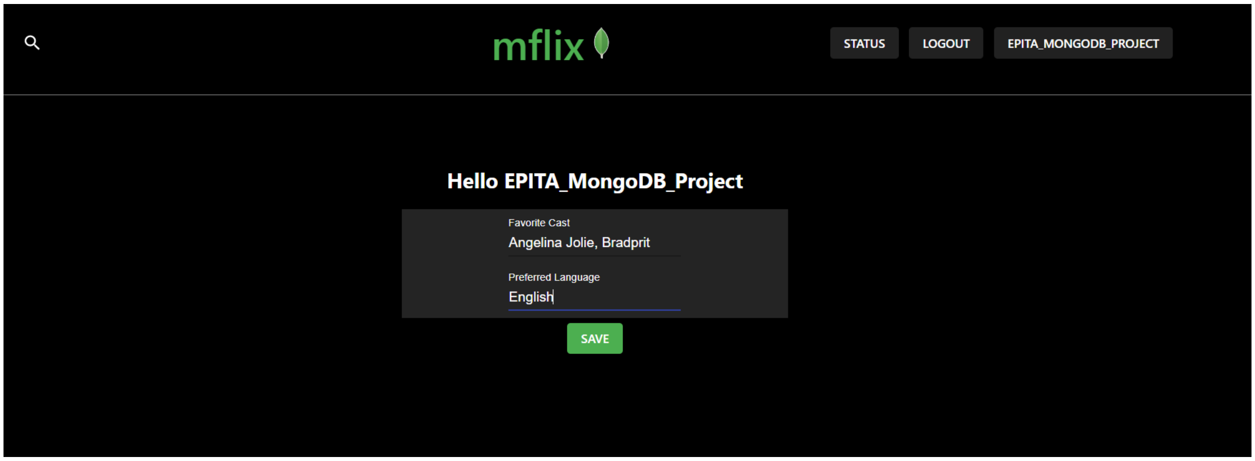
Only Titles are displayed – “titles”:1



```
def get_movies_by_country(countries):
    """
    Finds and returns movies by country.
    Returns a list of dictionaries, each dictionary contains a title and an _id.
    """
    try:

        """
        Ticket: Projection

        Write a query that matches movies with the countries in the "countries"
        list, but only returns the title and _id of each movie.

        Remember that in MongoDB, the $in operator can be used with a list to
        match one or more values of a specific field.

        """
        #cursor = db.movies.find( { "countries": {"$in":[countries]} }, { "title": 1,"_id":0 } )
        #from bson.json_util import dumps
        #print(dumps(cursor, indent=2))

        # TODO: Projection
        # Find movies matching the "countries" list, but only return the title
        # and _id. Do not include a limit in your own implementation, it is
        # included here to avoid sending 46000 documents down the wire.
        return list(db.movies.find({"countries": {"$in": countries}},{"title": 1}))

    except Exception as e:
        return e
```

- ## UPDATE

Adding the preferences of the registered users. This update the users collections with new informations.



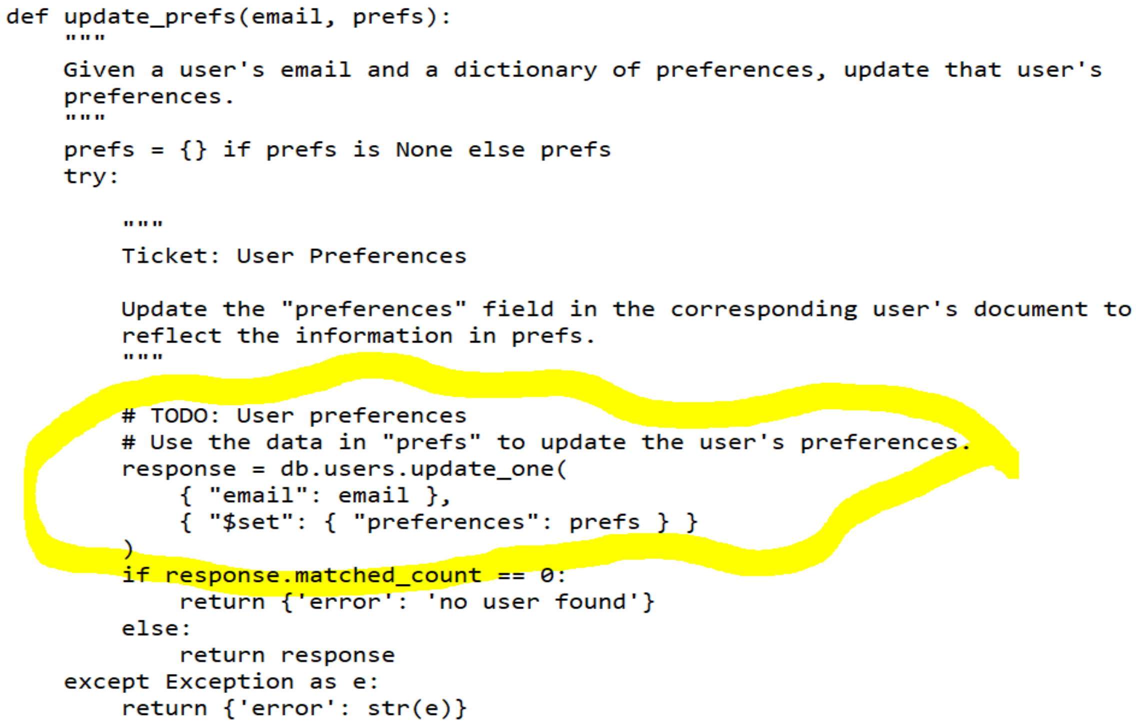
Hello EPITA_MongoDB_Project

Favorite Cast
Angelina Jolie, Bradprit

Preferred Language
English

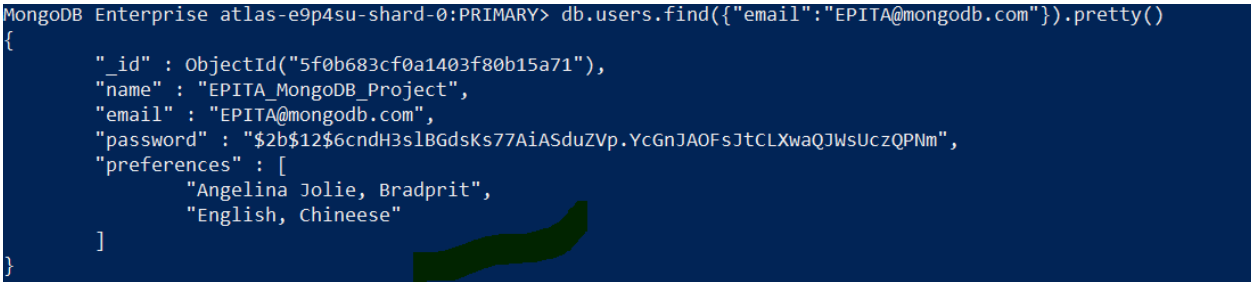
Using collections.update_one() from MongoDB

```
db.users.update_one(
        { "email": email },
        { "$set": { "preferences": prefs } }
    )
```



```
def update_prefs(email, prefs):
    """
    Given a user's email and a dictionary of preferences, update that user's
    preferences.
    """
    prefs = {} if prefs is None else prefs
    try:

        """
        Ticket: User Preferences

        Update the "preferences" field in the corresponding user's document to
        reflect the information in prefs.
        """

        # TODO: User preferences
        # Use the data in "prefs" to update the user's preferences.
        response = db.users.update_one(
            { "email": email },
            { "$set": { "preferences": prefs } }
        )
        if response.matched_count == 0:
            return {'error': 'no user found'}
        else:
            return response
    except Exception as e:
        return {'error': str(e)}
```

The details are updated in the users collection of sample_mflix database



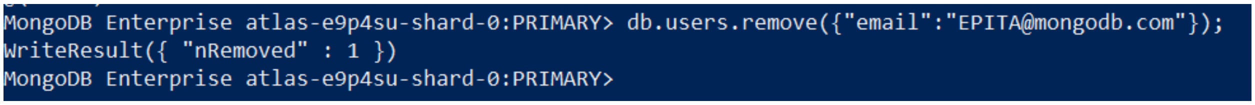
```
MongoDB Enterprise atlas-e9p4su-shard-0:PRIMARY> db.users.find({"email":"EPITA@mongodb.com"}).pretty()
{
        "_id" : ObjectId("5f0b683cf0a1403f80b15a71"),
        "name" : "EPITA_MongoDB_Project",
        "email" : "EPITA@mongodb.com",
        "password" : "$2b$12$6cndH3slBGdsKs77AiASduZVp.YcGnJAOFsJtCLXwaQJWsUczQPNm",
        "preferences" : [
                "Angelina Jolie, Bradprit",
                "English, Chineese"
        ]
}
```

- ## DELETE

Delete the details from database.

Using collections.remove() from MongoDB

db.users.remove({ "email": email })



```
MongoDB Enterprise atlas-e9p4su-shard-0:PRIMARY> db.users.remove({"email":"EPITA@mongodb.com"});
WriteResult({ "nRemoved" : 1 })
MongoDB Enterprise atlas-e9p4su-shard-0:PRIMARY>
```

The user is deleted successfully deleted from the database.

References

- https://university.mongodb.com/
- https://docs.mongodb.com/
- https://www.guru99.com/mongodb-tutorials.html
- https://discourse.university.mongodb.com/c/M220P/6
- https://github.com/jkcg-learning/MFlix